A Construction of Locality-Aware Overlay Network: mOverlay and Its Performance

Xin Yan Zhang, *Student Member, IEEE*, Qian Zhang, *Member, IEEE*, Zhensheng Zhang, *Senior Member, IEEE*, Gang Song, and Wenwu Zhu, *Senior Member, IEEE*

***Abstract*—Recently, there are many research interests in the peer-to-peer overlay architectures. Most widely used unstructured peer-to-peer (P2P) networks rely on central directory servers or massive message flooding, clearly not scalable. Distributed hash tables (DHT)-based structured overlay networks are expected to eliminate flooding and central servers, but can require many long haul messages deliveries. Thus, one important aspect of constructing an efficient overlay network is how to exploit network locality in the underlying network. In this paper, we propose a novel mechanism, mOverlay, for constructing an overlay network that takes account of locality of network hosts. The constructed overlay network can significantly decrease the communication cost between end hosts by ensuring that a message reaches its destination with small overhead and highly efficient forwarding. To construct the locality-aware overlay network, dynamic landmark technology is introduced. In this paper, we first present an effective locating algorithm for a new host joining the overlay network. Furthermore, we present theoretical analysis and simulation results to evaluate the network performance. Our analysis shows that the overhead of our locating algorithm is $O(\log N)$, where N is the number of hosts in the overlay network. Our simulation results show that the average distance between a pair of hosts in the constructed overlay network is only about 11% of the one in a traditional, randomly connected overlay network. Network design guidelines are also provided in the paper. Many large-scale network applications, such as media streaming, application-level multicasting, and media distribution can leverage our proposed locality-aware overlay, mOverlay, to enhance their performance.**

***Index Terms*—Distributed algorithms, overlay network, quality-of-service (QoS).**

I. Introduction

MANY emerging peer-to-peer (P2P) networks, such as Gnutella [1], Freenet [2], Napster [3], and Morpheus [4], have the common feature that all the hosts in the P2P network form an overlay network and exchange information through the overlay. Several next-generation P2P systems (CAN [6], Chord [7], Pastry [8], and Tapestry [9]) provide a self-organized overlay to enhance the search performance.

Manuscript received October 9, 2001. The work of X. Y. Zhang, Z. Zhang, and G. Song was done while the authors were with Microsoft Research Asia.

X. Y. Zhang is with The Chinese University of Hong Kong (CUHK), Hong Kong (e-mail: xyzhang2@ie.cuhk.edu.hk).

Q. Zhang and W. Zhu are with Microsoft Research Asia, Beijing Sigma Center, Beijing 100080, China (e-mail: qianz@microsoft.com; wwzhu@microsoft.com).

Z. Zhang is with San Diego Research Center, San Diego, CA 92130 USA (e-mail: zzhang@ieee.org).

G. Song is with the Department of Computer Science and Technology, Tsinghua University, Beijing, China (e-mail: songgang97@mails.tsinghua.edu.cn).

Digital Object Identifier 10.1109/JSAC.2003.818780

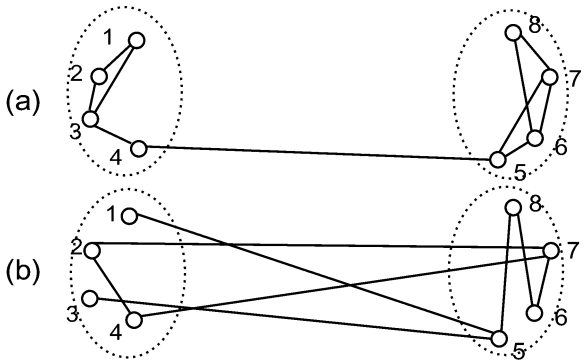


Fig. 1. (a) Illustration for locality-aware overlay and (b) randomly connected overlay.

In general, building an overlay should provide flexibility and enable rich services. However, one of the crucial issues in deploying an overlay network is the potential performance degradation.

In an overlay network, data transfer might not be as efficient as the one performed at the network layer. Routing overhead is a key performance metric for overlay infrastructures. If the overlays are constructed randomly, nearby hosts in the overlay network may actually be far away in the underlying network. This may waste too much network resources and, therefore, degrade performance significantly. For example, in Fig. 1, host 1 and host 4 are transferring messages between them. In the randomly connected overlay [Fig. 1(b)], the path used to transfer the control messages is 1, 3, 6, 7, and 4. The links between hosts 3 and 6, and between hosts 7 and 4 are very long. This is obviously inefficient. It is desirable to construct an overlay as in Fig. 1(a). Therefore, this raises the requirement on the construction of an efficient overlay network to achieve better performance with low overhead and low extra network traffic. In particular, we focus on the following two aspects in the overlay construction.

1) *Efficiency*: An overlay construction scheme should make sure that the communication on the overlay is not too costly, compared with the underlying network. Based on this principle, the neighboring hosts in the overlay should be closely located in the underlying network. To achieve this goal, one should take the node locality into account. That is, the protocol should be aware of the location of a host to be added to the overlay. This can reduce the delivery time and consequently the cost of communications.
2) *Scalability*: The overlay network should remain tractable with the increasing number of hosts and data traffic. The

cost of the overlay network maintenance, which includes data management and locality management, should be as small as possible. The overlay construction process should be implemented in a distributed fashion. Reducing the number of long-distance hops traversed per overlay route and reducing the bandwidth usage in constructing the overlay network will improve the scalability of the overlay network.

Based on the above observations, it can be seen that the locality or network proximity is an essential characteristic for the overlay construction, since it will affect the efficiency and consequentially the scalability of the designed overlay.

Fully decentralized systems such as Gnutella [1] and Kazaa [5] are studied by many researchers. These P2P systems are unstructured in that the overlay topology is ad hoc and the placement of data is completely independent of the overlay topology. However, to the best of our knowledge, there are very few overlay networks in which the node locality is taken into account. Some "highly structured" P2P overlays, such as CAN [6], Chord [7], Pastry [8], and Tapestry [9], are designed to enhance the searching performance. However, as pointed out in [19], structured designs are likely to be less resilient in the face of a very transient user population, because it is hard to maintain the structure (such as neighbor lists, etc.) required for routing to function efficiently when hosts are joining and leaving at a high rate. Chord in its original design, for instance, does not consider network proximity at all. As a result, its messages may travel arbitrarily long distances in the overlay network in each routing hop. Some modification to CAN, Pastry, and Tapestry are made to provide locality to some extent. However, these results come at the expense of a significantly more expensive overlay maintenance protocol, compared with Chord.

In [28] and [29], the authors also advocate the concept of topology-awareness. However, their method is directly using the border gateway protocol (BGP) routes information. For overlay network which built on top of application layer, it might not be practical to use that kind of information. In [10], the authors propose a topology aware system based on landmarks, similar to our work, trying to meet the efficiency requirement. Although the efficiency can be improved in [10], this solution needs extra deployment of landmarks and produces some hotspots in the underlying network when the overlay is heterogeneous and large.

Application layer multicast algorithms construct a special overlay network exploiting network proximity. The protocol they used are often based on a tree structure or a mesh structure. Mesh structure such as Narada [26] is suitable for small overlay, often considered not scalable. In the tree structure such as NICE [16] and HMTP [27], a new node who wants to join the overlay will first contact the highest level host. Thus, the highest level host becomes a hot spot. This hierarchical system is vulnerable in nature because higher level hosts will be more important in the architecture. When those higher level hosts fail, the system may be unstable and need rather long time to recover.

In this paper, we construct an overlay network by exploiting the locality in the underlying network using the group concept. The group concept introduced in this paper does not rely on static landmarks; therefore, our proposed solution has higher scalability and robustness. With the group concept, the maintenance cost for our proposed overlay is significantly reduced. Meanwhile, the locating complexity is also rather low. Using the locating scheme introduced in the host-join protocol, a new host can find its nearby groups within $O(\log N)$ steps, where N is the number of hosts in the overlay network. With the dynamic landmarks used in the overlay forming process, the load balance can be reached and hot spots can be avoided. Our simulation results indicate that the locality property is robust in a variety of network topology models. For example, the average distance between a pair of hosts in the constructed overlay network proposed in this paper is only about 11% of the one in a traditional, randomly connected overlay network. Many applications including multimedia distribution, file transferring, and application-level multicasting can leverage this locality-aware overlay for enhancing overall performance.

The rest of the paper is organized as follows. In Section II, we describe the basic model and protocols of our locality-aware overlay. Theoretical analysis and simulation results to evaluate the complexity of the locating algorithm and the efficiency of the overlay are given in Sections III and IV, respectively. Some application scenarios that can leverage our overlay are briefly introduced in Section V. Finally, Section VI concludes the paper.

II. Locality-Aware Overlay Construction

A. Basic Concept

Overlay networks are constructed by end-hosts. Each host in the overlay is running a protocol to communicate with other hosts. In general, each host maintains the information about a set of other hosts to communicate with. Two hosts are said to be neighbors if there is a connection (for communication purpose) through the overlay. If nearby hosts are constructed as neighbors and neighbor groups are connected, the maintenance messages/other application specific information delivery time will be significantly decreased. Therefore, a hybrid structure that is locality-aware is very desirable to increase the system performance.

Fig. 1 gives a high level illustration of what locality-aware overlay should look like. Given that the connectivity of each host in Fig. 1 is fixed, the problem is how to construct an overlay network so that the average neighbor distance is minimized. Randomly selecting neighbors [in Fig. 1(b)] may result in more links between far away hosts and fewer links between nearby hosts. The resulting average neighbor distance would be relatively long. A desirable locality-aware overlay structure [in Fig. 1(a)] is that most links are between hosts within a group and only one or two links between two groups. (For example, nodes 1 and 4 can communicate with each other through host 3 in Fig. 1(a), and through host 3, 6, 7 in Fig. 1(b), traveling two unnecessary long intergroup links. The advantage of the proposed locality-aware overlay is obvious). The proposed architecture is a two-level hierarchical network. The top level consists of groups and the bottom level consists of hosts within groups. Information is completely shared among all the hosts within one group.

Before describing the procedure of constructing a locality-aware overlay, we first introduce the concept of group. A group

consists of a set of hosts that are close to each other. For any positions P in the underlying network, if the distance between P and host A and the distance between P and host B are the same, we say that hosts A and B are in the same group. Here, the distance between two hosts can be network latency, or round trip time, or minimum bandwidth on the links along a path between the two nodes, or some user defined cost function between the two nodes.

A group can exchange messages with several other groups, which are referred to as neighbor groups. The neighbor groups in this overlay are the groups nearby in the underlying physical network.

In deciding which group a new host should belong to, we introduce the following criterion.

Grouping Criterion: When the distance between a new host Q and group A's neighbor groups is the same as the distance between group A and group A's neighbor groups, then host Q should belong to group A.

Here, the neighbors of a group are acting as the ***dynamic landmarks*** (used in the grouping criterion). We choose the nearest groups to be such landmarks because they are accurate for measurement compared with faraway groups. Another advantage to have nearby neighbors is that the measurement and the data transfer will have less overhead cost than faraway neighbors. This can significantly decrease the total cost for maintenance of the overlay, and is beneficial in many applications.

B. Basic Components for Overlay Construction

In forming the overlay, each group is a self-organizing cluster of hosts. When a new host arrives, it uses a locating method to join a nearest group or form its own group according to the grouping criterion.

There are mainly two parts in the entire framework. One is the algorithm to find a nearby group for a host to join, i.e., to route to the nearest group. It is obvious that the performance of the overlay highly depends on the efficiency of this nearby group locating process, details of which are described in Section II-C. The other part is a set of formation and maintenance protocols of this locality-aware overlay. Once a host in a group obtains network performance measured to one remote host in another group by active probing, it disseminates that information to all other hosts in the same group if needed. Each group will maintain a local host cache, which contains H hosts in its group. These hosts are responsible for communication with hosts in other groups. The group also maintains information about its M neighbor groups, such information may be the distances, or hosts in their neighbor groups' host cache, etc. The first host in the host cache, called leader, is responsible for the information update for the host cache and neighbors. Details of our protocols are given in Section II-D.

C. Locating Process

In the following, we present the detailed locating algorithm, which is based on distance measurement, such as [13] or IDMaps in [14]. The algorithm can be considered as a procedure of routing to the nearest group. It is assumed that there is a global host cache called the rendezvous point (RP) in the overlay network. All new hosts know where the RP is. RP is usually a machine or a set of machines, which give the new host the start point in the overlay. Therefore, the first step for a host to join the network is to contact RP to fetch several hosts. These hosts, called boot hosts, will guide the new host to the nearest group. As illustrated in Fig. 2, a new host A first communicates with an RP. The RP informs host A to contact a boot host B in group 1. Host A then measures the distance from itself to host B. At the same time, host B sends all the information about the neighbor groups of group 1 to host A. All those neighbor groups are stored in a list, which is referred to as candidate group list. Host A will select groups sequentially from the candidate list, measure the distance between itself and the selected group, and determine the closest group. If the grouping criterion is met, host A belongs to group 1 and the process terminates. Otherwise, the current closest group to host A and the current shortest distance, denoted by $D_{\min}$ are recorded. In Fig. 2, group 2 is the closest to host A. A boot host in group 2 will in turn send all the information about its neighbor groups to host A. The distances between host A and all the neighbor groups of group 2 will be measured. The grouping criterion is checked again. If met, host A belongs to group 2 and the process terminates. Otherwise, the shortest distance among the neighbor groups of group 2 to host A, say from host A to group 3, will be found and compared with $D_{\min}$. If $D_{\min}$ is greater, $D_{\min}$ will be replaced by the newly obtained shortest distance and group 3 will be remembered as the current group which has the shortest distance to host A. Otherwise, if $D_{\min}$ is smaller, group 2 is still the current group which has the shortest distance to host A. The same procedure will start at group 3 and be repeated until the grouping criterion is met. If the grouping criterion has not been met after visiting all the groups, the recorded group, which has the distance $D_{\min}$, will be chosen as the closest group. In Fig. 2, host A will join group 6. Note that if it takes a long time to visit all the groups, it is necessary to set up a stop criterion so that the locating algorithm will terminate within a given time period. One criterion is to stop the algorithm after repeating the procedure C times. The prototype of the algorithm is illustrated below.

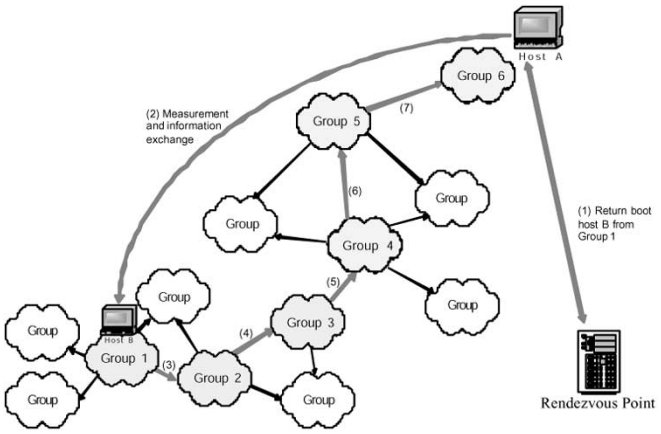


Fig. 2. Illustration of the locating process.

```
Group_Locating(host)
{
  Boothost = Get_Boot_Host(RP)
  Group = Get_Group(Boothost)
```

```
  do {
    CandidateList = Get_Neighbors(Group)
    DistanceList = Measure_Distance(host, CandidateList)
    if (Meet_Group_Criterion (Group, DistanceList))
      return Group
    (MinDis, MinGroup) = Min(DistanceList)
    if (MinDis < CurrentMinDis)
    {
      CurrentGroup = MinGroup
      CurrentDis = MinDis
    }
    if (Meet_Stop_Criterion())
      return CurrentGroup
    Group = MinGroup;
  } while (true);
}.
```

From the algorithm described above, we can see the first group is randomly selected by RP, and the new hosts should have equal probability to be in one of the existing group. So we may safely assume that each group has equal probability to be visited by the new hosts. Therefore, this dynamic landmarks algorithm can provide load balance for each group.

D. General Overlay Operations

In this section, several protocols of the overlay formation and maintenance will be described. These protocols are similar to the ones in traditional unstructured overlay. However, we make several modifications to the existing protocols so that better scalability and robustness can be achieved in our framework.

1) Forming New Group: In the initialization stage of overlay network formation, new groups are more frequently generated since few groups exist. Another situation that a new group should be formed is when the nearest group for the new host does not meet the grouping criterion. The procedure of forming a new group is simple. More specifically, the host declares a new group with a group ID, at the same time, finds its M neighbors.

To find its M neighbors, the first step is to repeat the locating procedure from several boot hosts given by RP. Several nearby groups are obtained for this new host. The host then obtains neighbor information from its nearest neighbor. Due to the locality characteristic of nearby groups, one group's neighbor will probably in turn be the neighbor of its neighbors. By iteratively repeating this procedure, one can find most of its neighbors from the first neighbor.

2) Group Joining: In a normal case, if a host has located its proper group, the new host joins that group and directly connects to several hosts in the group. Meanwhile, from these hosts, the new host can get the information for group maintenance.

3) Information Sharing: Information sharing is an important component of group operation. Since hosts in the same group has similar network characteristics, the distance from a host A to a specific group X can be regarded as the distance from all the hosts in A's group to X. Information about the measurement performed between host A and group X should be sent to all other hosts in A's group, with low overhead. Based on the nature of our constructed overlay, information within a group can be shared through flooding. Due to the fact that hosts within a group are nearby, the broadcast overhead should not have a significant impact on the underlying network.

4) Information Updating: In our overlay network, information updating is a distributed algorithm run by each host of the group. A local host cache is used to complete the updating task. All the hosts in a group maintain the same host cache, which has H hosts, the hosts in the cache will take the responsibilities in its natural sequential order. That is, the first host in the host cache acts as a leader, which is responsible for information update for the whole group. To indicate its existence, the leader periodically sends out "alive" messages to the whole group. If the leader fails, the hosts in the group will know after a period of time as they do not receive the "alive" message from the leader. At this moment, the second host in the list will automatically be the leader and send out its "alive" messages. If the second also fails, the third one will stand up. Therefore, if not all the hosts in the local host cache fail, the group operates normally. It is obvious that the probability of all H hosts fail within a given time period is extremely small if H is large enough. Note that the host cache needs to be synchronized only if there are changes occurred. No global synchronization is needed.

If, unfortunately, all H hosts in the host cache fail, after H periods, no leader appears. Under this situation, we propose a solution based on time stamp. At that time, a host, who realizes that all H hosts fail, will send a message to declare its leadership. Since all the hosts may notice the same phenomenon, they may send messages at the same time. There are many ways one can select a leader. For example, one can add a time stamp in the message and select the host whose message has the earliest time stamp as the leader. This will be a unique choice. The new leader will rebuild the host cache for the group. Or one can add some metric such as the speed of the machine at the node in the message and select one which is the fastest. Other criteria are also possible.

In our architecture, the information that needs to be updated is divided into two categories. One is the host cache and the other is the neighbors of groups. Host cache can be updated when a new host joins. When the hosts in the host cache receive the "alive" messages from the leader, each of them will send a reply to the leader to indicate its existence. A fresh host, assigned by the leader in the group, will replace failed host. This procedure will keep the hosts in the local host cache alive. Any changes in the host cache will be sent to all the hosts in the group and to its neighbor groups.

The information of neighbor groups will be updated when a nearby group is generated. The leader of the new group, generally the first host of the group, will probe the distance to its nearby group. The nearby group will also be aware of the distance and decide if its neighbors list should be refreshed. If so, the leader will inform all the hosts in its group about this neighbor information. It should be mentioned that the traffic due to update can be negligible for the following reason. The host cache contains only H host addresses, in which each address is typical 4 bytes. The total bytes will be $4(M+1)H$, assuming there are M neighbors. For example, if there are ten neighbors

and ten hosts in the host cache, the information needs to be transmitted is only 440 bytes.

5) Host Failure/Leaving: In our architecture, a host does not need to take any action when it leaves the overlay network. However, in order to improve system performance, when a host leaves, it is required that the host informs the leader or the second host if itself is the leader. The failure of a single host does not have any impact to its group, since the host cache will be updated periodically. Even if the host is the leader of the group, the group can automatically be aware of its departure and a new leader, the successor in the host cache, will appear as discussed in Section II-D4.

III. Overlay Performance Analysis

Having introduced the detailed operations, in this section, we give some quantitative analysis for the characteristics and performance of the constructed overlay. For convenience of analysis, the overlay will be considered as a graph, where each node in the graph represents a group in the overlay, neighbor relationship between two groups is represented by edges between two nodes in the corresponding graph.

A. Theoretical Approximation for the Searching Hops

As mentioned before, one key performance metric of our constructed overlay is the complexity of the algorithm to locate the nearby group. In the following, we give a theoretical approximation for the number of searching hops using the locating algorithm. Based on our definition of groups, in the graphical model, each node has out-degree of M if each group has, on average, M neighbor groups. The problem of finding a nearest group in the overlay is similar to finding a specific node from another node through edges in the graph.

One can prove that the average distance d between any two nodes follows the inequality

$$d < \log_M N + 3.$$

Therefore, the average distance could be considered roughly at the level of $\log_M N$. The proof is given in the Appendix. The approximation indicates that the new host could reach its nearest group with $O(\log N)$ communications.

Extensive simulation results given in the next section demonstrate that the approximate expression for the number of searching hops is accurate for several practical network topologies.

B. Efficiency

In this section, we evaluate the performance efficiency of our locality-aware overlay and compare with that of a randomly connected overlay. Here, we use the average neighbor distance of the overlay as one performance metric, as it reflects the overall maintenance cost and flooding-based searching overhead. In the rest of this section, we will theoretically calculate the average neighbor distance in our locality-aware overlay, denoted by $\bar{D}$, and in a randomly connected overlay, denoted by $\bar{D}'$, respectively. The simulation results for $\bar{D}$ and $\bar{D}'$ in several practical network topologies are given in the next section.

In the locality-aware overlay, suppose it contains N groups and each group consists of n hosts, on average. Every group has M neighbor groups. In every group, each host has m neighbor hosts. The average distance between neighbor groups is denoted as D_b and the average distance between two hosts in the same group is denoted as D_i. The total number of intragroup neighborhood links is $N \cdot n \cdot m/2$. The total number of intergroup neighborhood links is $N \cdot M/2$. Thus, the average neighbor distance can be represented as

$$\bar{D} = \frac{D_i \frac{N \cdot n \cdot m}{2} + D_b \frac{N \cdot M}{2}}{\frac{N \cdot n \cdot m}{2} + \frac{N \cdot M}{2}} = \frac{D_i \cdot n \cdot m + D_b \cdot M}{n \cdot m + M}. \quad (1)$$

In the following, we derive an expression for the randomly connected overlay. In such an overlay, there is no hierarchical structure. Any two hosts, whether or not they belong to the same group, have equal probability to be neighbors in the overlay. We can classify the entire neighborhood links into two categories, inter-group links and intra-group links. The underlying network of the randomly connected overlay is the same as that of the locality-aware overlay. We assume that the underlying network contains N groups and each group consists of n hosts, on average. Each host has m' neighbor hosts in the same group, and meanwhile has m'' neighbor hosts in all other groups. The average distances of intergroup links and intragroup links are denoted by D'_b and D'_i, respectively. Similar to the derivation of (1), we have

$$\bar{D}' = \frac{D'_i \cdot n \cdot m' + D'_b \cdot n \cdot m''}{n \cdot m' + n \cdot m''} = \frac{D'_i n \frac{m'}{m''} + D'_b n}{n \frac{m'}{m''} + n}.$$

In order to compute $\bar{D}'$, we need to calculate (m'/m'') first. As mentioned before, any two hosts in the underlying network have the same probability to become neighbors, which means the ratio of m' to m'' for each host equals the ratio of the total number of the hosts in the same roup to the total number of the hosts in all the other groups. That is

$$\frac{m'}{m''} = \frac{n-1}{(N-1)n}.$$

Thus, the average neighbor distance in the random overlay can be derived as follows:

$$\begin{aligned} \bar{D}' &= \frac{D'_i n \frac{m'}{m''} + D'_b n}{n \frac{m'}{m''} + n} \\ &= \frac{D'_i (n-1) + D'_b n (N-1)}{nN-1} \\ &= \left[\frac{D'_i (n-1)}{D'_b (nN-1)} + \frac{n(N-1)}{nN-1} \right] \cdot D'_b. \end{aligned} \quad (2)$$

Noting that $N >> 1$ and $D'_i << N D'_b$ (the distance within a group is much smaller than N times the distance between groups), (2) can be approximately written as $\bar{D}' \approx D'_b$.

Now, from (1), we have an approximation for the ratio of $\bar{D}$ to $\bar{D}'$

$$\frac{\bar{D}}{\bar{D}'} \approx \frac{D_b}{D'_b} \frac{1}{n \cdot \frac{m}{M} + 1} \left(\frac{D_i \cdot n \cdot m}{D_b \cdot M} + 1 \right). \quad (3)$$

Notice that D'_b and D'_i are fixed and determined by the underlying network, D_b, and D_i depend on the overlay construction process and can be obtained through simulation, (m/M)

is the ratio of the number of neighborhood links within groups to number of links between groups, and n is the group size. A desirable constructed overlay should have $D_i \approx D_i'$ because whether within a group or not, neighbors are selected randomly and similarly in both locality-aware overlay and randomly connected overlay.

The importance of the ratio $\bar{D}/\bar{D}'$ can be understood as follows. The smaller the ratio is, the shorter the average neighbor distance in the locality-aware overlay is (compared with that of the randomly connected overlay). The goal of constructing a locality-aware overlay is to fully utilize the locality property of the underlying network so that the average neighbor distance is as small as possible. Some of the parameters, such as D_b' and D_i' in (3) are fixed, some parameters, such as m, D_b, and D_i, depend on the construction procedure. Some observations can, therefore, be made from (3). In order to increase the efficiency (decrease the ratio) of the overlay, we should try to:

- shorten the neighbor links between different groups (to decrease D_b);
- have more neighbors in the same group if possible (to increase m);
- always place hosts to their proper group if possible (to decrease D_i so that $D_i \approx D_i'$).

The above analysis can be served as an overlay design guideline.

C. Robustness

In our proposed overlay, due to the locality characteristic, the connectivity of the graph might be an issue. All of the methods discussed in the above section cannot find the nearest group for the new host if the underlying network has several isolated subnetworks (in such case; there is no path in the overlay from the targeted group and the boot host). To prevent this from happening, in our protocol, each group will select a random group as its special neighbor group; this will significantly decrease the probability of disconnected graph.

In the hybrid structure we proposed, the disappearance of any host in the overlay almost does not do any harm to the whole network. The local host cache can provide good robustness for the network.

D. Scalability

The maintenance cost of our hybrid structure can be seen as an advantage comparing with mash-based or tree-based overlays. In large overlay such as current P2P network, there might be millions of hosts. At that environment, mesh-based structure cannot survive because the network cannot afford $O(N^2)$ communications between hosts. In a tree-based structure, the upper hosts, especially the root will endure large load. For example, in an overlay network with 800 000 hosts, suppose in average each hosts has an online time of one hour. The root will have $800\,000/3600 = 222$ join requests/second. For a normal desktop, this is a rather high workload and also will cost quite some bandwidth. In that case, the root, which is the most crucial point to the whole overlay, will be quite vulnerable. However, with our dynamic landmark location algorithm, the 222 join requests will be distributed to all the groups evenly. Including joining hops, each group will process less than ten join requests per second. This is quite reasonable for any host in an overlay. In that point, we can safely say that our algorithm is more scalable than tree-based or mesh-based algorithms.

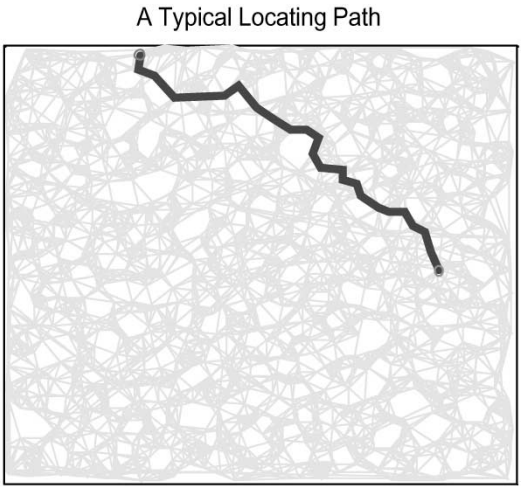


Fig. 3. A typical locating path from circle on the top to circle on the right.

IV. Simulation

In this section we carry out several simulations to quantify the performance of new host joining algorithm. The simulation results show that new host can quickly locate the nearby group to join and construct an efficient overlay.

A. Simulation Methodology

In each simulation, we first use a generated network topology map as the underlying network. Each node in the map represents a local-area network (LAN) consisting of 100 end hosts. The size of the network varies from 50 to 800 000. When a new host arrives, it first selects a starting group through RP, from which it iteratively locates the nearest group to join. The simulator will find a path from the starting group to the final group iteratively.

To evaluate how fast a new host can join the overlay, we calculate the hops from the starting group to the final group. To evaluate the quality of the overlay (compared with that of a randomly connected overlay), we calculate the average neighbor distance of the overlay.

B. Network Topology

Three network topology models are used in the simulation. A Euclidean model is a hypercubic in a D-dimension Euclidean space. Nodes are randomly located in the hypercubic. The distance metric corresponds to the Euclidean distance metric between the nodes. Using this model, we can generate topology maps with easy control of parameters and plot the joining process to verify the algorithm. Fig. 3 shows a typical joining path in a two-dimensional (2-D) Euclidean model map.

The other two network topology models used in the simulation are Barabasi–Albert [23] and Waxman [22] models, which are used by well-known Internet topology generators. Waxman model deals with general random networks. In Barabasi–Albert model, the nodes are interconnected with preference to higher degree nodes. This model produces the power law distribution of

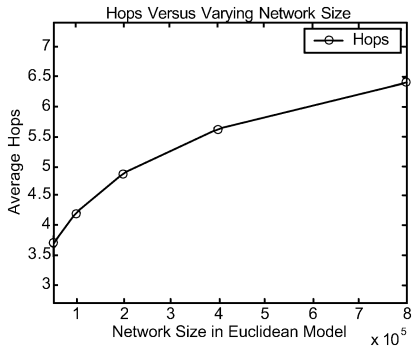


Fig. 4. Performance versus network size N.

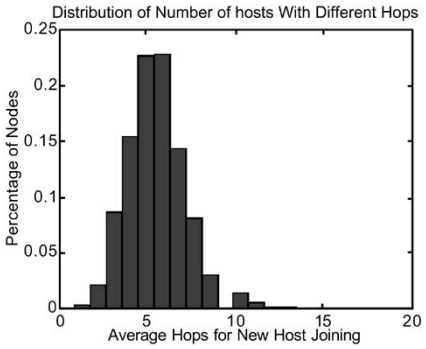


Fig. 5. Distribution of the number of hosts with different hops.

connections. In other words, we use Inet as a power-law model. The topology maps of the two models are derived from BRITE [15]. The parameter we used in Barabasi–Albert and Waxman models is $m = 2, \mathrm{alpha} = 0.19, \mathrm{beta} = 0.2$. More details about the parameters in BRITE can be found in [15].

C. Simulation Results on Euclidean Models

There are many parameters that influence the network conditions and locating process performance, of which the most important ones are the following:

- N: total number of hosts in the network;
- N_g: the number of neighbors a group maintains.

To evaluate the performance versus network size, we simulate the group joining procedure in different networks, with sizes from 50 to 800 000, and count the number of hops to join the group for a randomly selected node.

Fig. 4 shows the average hops versus the total number of hosts, N. We can see that the hops increase in the order of $O(\log N)$as N increases. Therefore, the cost of the joining process can be achieved in $O(\log N)$ hops, which is in agreement with the analytic results in Section III-B.

In Fig. 5, we show the distribution of the number of hosts with different number of hops to join. The data are collected using a Euclidean model network topology consisting of 400 000 hosts. It can be shown that more than 90% of all the hosts can join their groups in less than eight hops.

Next, we investigate the relation between N_g and the average number of hops to join. It can be seen from Fig. 6 that smaller N_g has more impact on the locating performance than larger N_g. However, after N_g increases to a certain value, its impact will

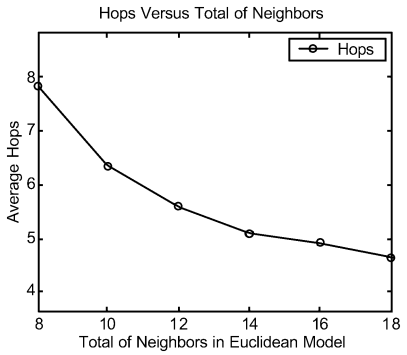


Fig. 6. Performance with varying Ng.

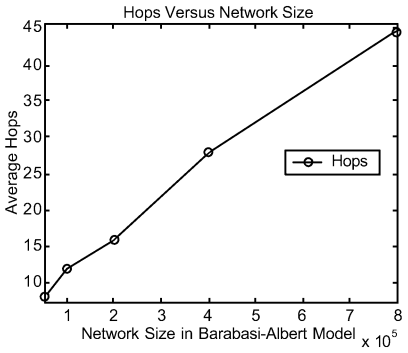


Fig. 7. Performance versus network size in Barabasi–Albert model.

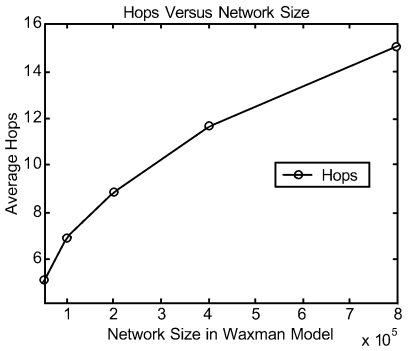


Fig. 8. Performance versus network size in Waxman model.

be reduced. This observation is useful in designing an overlay network.

D. Simulation Results on Barabasi–Albert Model and Waxman Model

We generated 10 topology maps of Waxman model and Barabasi-Albert model using BRITE group-level generator. 100 hosts are added to each group. The five maps of each model consist of 50, 100, 200, 400, and 800 thoursand hosts, respectively. The link (between group) delays are obtained by BRITE. The distances between any two hosts are calculated based on the shortest path. In all the simulations, each group keeps 14 neighbor groups.

Figs. 7 and 8 illustrate the average number of hops to join versus network size in Barabasi–Albert model and Waxman model, respectively. It can be seen that the algorithm runs better for Waxman than Barabasi–Albert model.

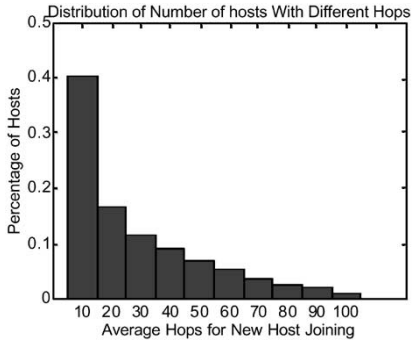


Fig. 9. Distribution of the number of joining hosts with different hops in Barabasi–Albert model.

TABLE I
THE PERCENTAGE OF JOINING HOSTS WITH CERTAIN UPPER BOUND OF HOPS IN BARABASI–ALBERT MODEL

network size	50K	100K	200K	400K	800K
Percentage %	86.4	91.0	91.7	86.1	80.6
upper bound of hops	16	28	40	52	64

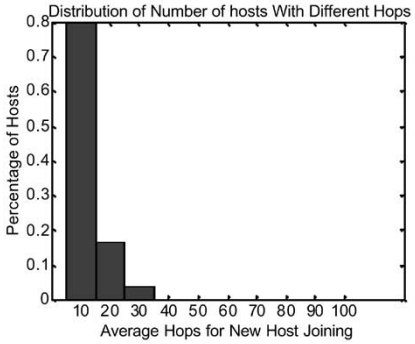


Fig. 10. Distribution of the number of joining hosts with different hops in Waxman model.

Fig. 9 shows the distribution of the number of hosts with different number of hops to join in the Barabasi–Albert model. The data are collected using a network topology consisting of 800 000 hosts. It can be seen from this figure that most hosts can find the right group to join within a small number of hops. Table I gives the percentage of hosts joining the right group with corresponding upper bound on the number of hops for different network sizes. The corresponding results obtained from Waxman model are given in Fig. 10 and Table II.

E. Comparison With Randomly Connected Overlays

In this section, we compare the performance of our proposed overlay with that of randomly connected overlays. Assume that the underlying network is the same for both overlays, and the number of edges and nodes are also the same in both overlays. It is assumed that the average distance between hosts in one group is only 10% of the average distance between different groups. The average neighbor distance in the locality-aware overlay is computed from (1), where D_b and D_i are obtained through simulation. The average neighbor distance in the randomly connected overlay is computed from (2) directly.

TABLE II
THE PERCENTAGE OF JOINING HOSTS WITH CERTAIN UPPER BOUND OF HOPS IN WAXMAN MODEL

network size	50K	100K	200K	400K	800K
percentage %	86.8	90.0	91.8	92.7	90.9
upper bound of hops	8	16	22	28	32

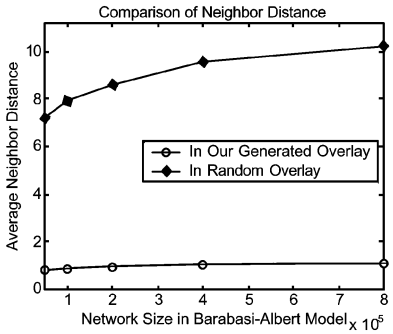


Fig. 11. Average distance of neighbor links in Barabasi–Albert model.

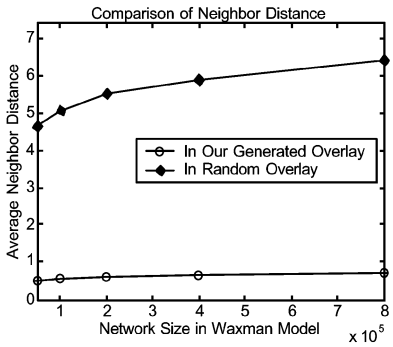


Fig. 12. Average distance of neighbor links in Waxman model.

Figs. 11 and 12 show the results derived in (1) and (2) for the Barabasi–Albert model and the Waxman model, respectively. The average ratio (over all the network sizes in Figs. 11 or 12) of the average neighbor distance of our generated overlay to that of the random connected overlay is 10.9% for Barabasi–Albert model and 11.0% for Waxman model, respectively. These small ratios indicate that our locality-aware overly indeed fully utilize the locality of the underlying network.

V. APPLICATION SCENARIOS

Many upper-layer applications can be leveraged using this locality-aware overlay network to improve their performance. In this section, we briefly discuss several application scenarios of the overlay network.

A. Application-Level Routing

File sharing is one of the most widely used applications for P2P overlay network. Without network proximity, users often experience extreme long file transfer delays caused by overloaded link or improper physical routing [12]. Using our generated overlay, the data can be transferred through more efficient logical paths by avoiding unnecessarily long/busy links. The corresponding transferring time can be significantly reduced.

B. Application-Level Multicast

Multicast in large scale networks is a hard problem for Internet service providers. Our overlay is a more desirable platform for application-level multicast since it fully utilizes the topology information to form groups. Multicast at group level would reduce the network traffic dramatically compared with no-hierarchic networks. Based on our overlay, some efficient algorithms can be proposed.

C. Media Distribution

Multimedia content distribution is similar to file sharing, but one key difference is that the quality-of-service (QoS) requirements for multimedia content delivery are much stringent than file sharing. A well-designed framework based on our overlay is described in [24], which achieves better performance compared with several existing approaches.

VI. Conclusion and Future Work

In this paper, we present a new protocol to generate an unstructured overlay to achieve locality characteristic. To decrease the communication cost between end hosts, we build a two-level overlay with network proximity. Specifically, a low overhead locating algorithm is proposed to form the overlay with locality characteristic. One of the key characteristics our overlay model has is that the overlay is constructed using dynamic landmarks (versus other existing approaches that use fixed landmarks). We can achieve locality-aware in unstructured overlay network without relying on the static landmark. Therefore, our solution has strong scalability and high robustness. Meanwhile, the locating complexity is also rather low. Our analysis and simulation results show that: 1) this simple locating algorithm will limit the overhead in the level of $O(\log N)$ and 2) the performance of our overlay improves significantly comparing with the one in traditional unstructured overlays. Our work is a significant step toward better understanding and using of unstructured overlays for applications over large-scaled underlying networks.

We are currently deploying this protocol into real network environments. We are also exploring some exciting usage of this locality-aware overlay, such as media streaming, application-level multicasting, and media distribution.

Appendix
Complex Analysis of the Locating Process

In this section, we study the average distance between any two nodes. An exact expression is very difficult to obtain and instead an upper bound on the average distance will be given. The locality-aware overlay constructed in the previous section can be thought as a special graph with N nodes (each node corresponds to a group in the constructed overlay), and each node has a fixed number M of neighbors selected randomly from the graph.

To derive an expression of the average distance between node A, chosen arbitrarily, and any other node in the graph, we consider the graph as a tree structure. The root of the tree is node A (Fig. 13. level 0).

Node A's direct neighbors, B, C, D are one-hop away and are placed below node A (at level 1). Here, we call A as the parent of nodes B, C, D. And nodes B, C, and D are children

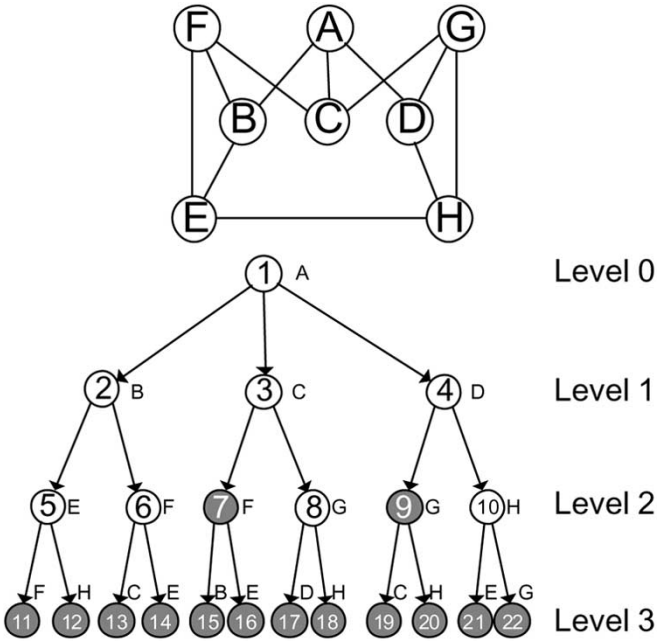


Fig. 13. Construction of the tree used in the proof.

of node A. In turn, each node adds its neighbors into the tree. A node, however, cannot add its parent into the tree (since it is already in the upper level). The process is repeated till all the nodes are visited. In the tree, we say a node is old if this node already exists in the tree. Otherwise, the node is new. For example, when the seventh node (node F, shadow one) is added into the tree by node C, the node F is already in the tree and, therefore, is old. For convenience of analysis, we assume that a node has $M+1$ neighbors. Any node in level $i, i > 0$, has M children (one neighbor is the node's parent) and node A has $M+1$ children.

It is apparent that the distance from node A to a node at level k is k. Let $g(k)$ denote the number of unique nodes in level k which have not appeared in upper levels. Let $K_{\max}$ denote the number of possible levels. The average distance between node A and any other node is then given by

$$d = \sum_{k=1}^{k} \frac{k \cdot g(k)}{N}. \tag{A1}$$

From (A1), if we know $g(k)$, the average distance d can be obtained easily. In the following, we derive an expression for $g(k)$. Please note, due to the fact that a node can be neighbor of several other nodes, at each level, one node could be a child of several parent nodes. For derivation purpose, we number the nodes according to the sequence of the nodes entering the tree, even if a node is already in the tree, starting from node A (from level $i-1$ to level i, and within each level, from left to right). For example, in Fig. 13, nodes are numbered as $1, 2, 3 \ldots$. Please note that some nodes are given several numbers, for example, node F are 6 and 87, in Fig. 13. Let the last node's number, at level k, be t_k, then

$$t_k = \frac{(M+1)M^k - 2}{M-1} \quad k = 0, 1, 2, 3 \ldots.$$

The total number of nodes at level k is $t_k - t_{k-1}$, some of the nodes are accounted several times. The actual number of unique nodes at level $k, g(k)$, is much smaller than $t_k - t_{k-1}$.

In the above tree, let V_t denote the tth node in the tree and $f(t)$ denote the total number of unique nodes in the tree after node V_t is added to the tree. Then, $g(k) = f(t_k) - f(t_{k-1})$. In the following, we derive expressions for $f(t)$. By definition, we have

$$f(t) - f(t-1) = \delta_t \tag{A2}$$

where $\delta_t = 1$ if node V_t is a new node and $\delta_t = 0$, otherwise.

Let $X(t)$ denote the probability that V_t is a new node, that is $X(t) = \mathrm{P}(\delta_t = 1)$. Before node V_t joins the tree, there are $N - f(t-1)$ nodes which are not in the tree. Since all the nodes are chosen randomly, thus, we have

$$X(t) = \frac{N - f(t-1)}{N}.$$

Let $\bar{f}(t)$ be the expected value of function $f(t)$ and taking the expectation on both sides of (A2), we have

$$\bar{f}(t) - \bar{f}(t-1) = \frac{N - \bar{f}(t-1)}{N}$$

or

$$\bar{f}(t) = 1 + \frac{N-1}{N}\bar{f}(t-1) \qquad t = 1, 2, \ldots, \text{and } f(0) = 0.$$

Solving the above equation iteratively, and noting that $f(0) = 0$, we obtain

$$\bar{f}(t) = N\left(1 - \left(\frac{N-1}{N}\right)^t\right) \approx N\left(1 - e^{-\frac{t}{N}}\right)$$

for N sufficient large.

Based on the definitions of $g(k)$ and $f(t)$, we have $g(k) = \bar{f}(t_k) - \bar{f}(t_{k-1})$, and

$$d = \sum_{k=1}^{K_{\max}} k\left(\frac{\bar{f}(t_k) - \bar{f}(t_{k-1})}{N}\right).$$

Since $\lim_{k\to\infty} k(1 - (\bar{f}(t_k)/N)) \to 0$

$$\begin{aligned} d &\le \sum_{k=1}^{\infty} k\left(\left(1 - \frac{\bar{f}(t_{k-1})}{N}\right) - \left(1 - \frac{\bar{f}(t_k)}{N}\right)\right) \\ &= \sum_{k=0}^{\infty}\left(1 - \frac{\bar{f}(t_k)}{N}\right) \\ &= \sum_{k=0}^{\infty} e^{-\frac{t_k}{N}}. \end{aligned}$$

The right-hand side of the above inequity is fairly complicated. In the following, we present a simple expression for the upper bound on the average distance. Dividing the summation into two parts, the first part is from $k = 0$ to $k = \mathrm{Log}_M N$ and the second part is from $k = \mathrm{Log}_M N + 1$ to infinity, as follows:

$$\begin{aligned} d &\le \sum_{k=0}^{\mathrm{Log}_M N} e^{-\frac{(M+1)M^k - 2}{(M-1)N}} + \sum_{k=\mathrm{Log}_M N+1}^{\infty} e^{-\frac{(M+1)M^k - 2}{(M-1)N}} \\ &< \mathrm{Log}_M N + 1 + \sum_{k=0}^{\infty} e^{-\frac{(M+1)NM^k - 2}{(M-1)N}} \\ &\le \mathrm{Log}_M N + 1 + \sum_{k=0}^{\infty} e^{-M^k} \end{aligned}$$

We know that when $M \ge 2$, $M^k \ge Mk$ and, hence, $e^{-M^k} \le e^{-Mk}$

$$\begin{aligned} d &\le \mathrm{Log}_M N + 1 + \sum_{k=0}^{\infty} e^{-Mk} \\ &= \mathrm{Log}_M N + 1 + 1/(1 - e^{-M}) \\ &< \mathrm{Log}_M N + 3 \end{aligned}$$

which completes the proof.

REFERENCES

[1] Gnutella Website. [Online]. Available: http://gnutella.wego.com
[2] Freenet Website. [Online]. Available: http://freenet.sourceforge.net
[3] Napster Website. [Online]. Available: http://www.napster.com
[4] Morpheus Website. [Online]. Available: http://www.musiccity.com
[5] Kazaa Website. [Online]. Available: http://www.kazaa.com
[6] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A scalable content-addressable network," presented at the ACM SIGCOMM, San Diego, CA, Aug. 2001.
[7] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," presented at the ACM SIGCOMM, San Diego, CA, Aug. 2001.
[8] A. Rowstron and P. Druschel, "Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems," presented at the 18th IFIP/ACM Int. Conf. Distributed Systems Platforms (Middleware 2001), Heidelberg, Germany, Nov. 2001.
[9] B. Zhao, J. Kubiatowicz, and A. Joseph, "Tapestry: An infrastructure for fault-tolerant wide-area location and routing," Univ. California, Berkeley, CA, TR UCB/CSD-01-1141, Apr. 2001.
[10] S. Ratnasamy, M. Handley, R. Karp, and S. Shenker, "Topologically-aware overlay construction and server selection," in *Proc. INFOCOM*, 2002, pp. 1190–1199.
[11] E. Ng and H. Zhang, "Predicting Internet network distance with coordiantes-based approaches," in *Proc. INFOCOM*, 2002, pp. 170–179.
[12] C. Labovitz, R. Malan, and F. Jahanian, "Internet routing instability," in *Proc. ACM SIGCOMM*, Sept. 1997, pp. 115–126.
[13] X. Zhang, J. Liu, Q. Zhang, and W. Zhu, "Measure: A group-based network performance measurement service for peer-to-peer applications," presented at the GLOBECOM, Taibei, Taiwan, 2002.
[14] P. Francis *et al.*, "IDMaps: A global internet host distance estimation services," *ACM/IEEE Trans. Networking*, pp. 525–540, Oct. 2001.
[15] BRITE. [Online]. Available: http://www.cs.bu.edu/brite/
[16] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, "Scalable application layer multicast," presented at the ACM SIGCOMM 2002, Pittsburgh, PA, Aug. 2002.
[17] E. Cohen and S. Shenker, "Replication strategies in unstructured peer-to-peer networks," presented at the ACM SIGCOMM 2002, Pittsburgh, PA, Aug. 2002.
[18] H. Tangmunarunkit, R. Govindan, S. Jamin, S. Shenker, and W. Willinger, "Network topology generators: Degree-based vs. structural," presented at the ACM SIGCOMM 2002, Pittsburgh, PA, Aug. 2002.
[19] Q. Lv, S. Ratnasamy, and S. Shenker, "Can heterogeneity make Gnutella scalable?," presented at the 1st Int. Workshop on Peer-to-Peer Systems (IPTPS), Cambridge, MA, 2002.
[20] P. Keleher, B. Bhattacharjee, and B. Silaghi, "Are virtualized overlay networks too much of a good thing?," presented at the 1st Int. Workshop on Peer-to-Peer Systems (IPTPS), Cambridge, MA, 2002.
[21] M. Castro, P. Druschel, Y. C. Hu, and A. Rowstron, "Exploiting network proximity in distributed hash tables," presented at the FuDiCo 2002: Int. Workshop on Future Directions in Distributed Computing, Bologna, Italy, June 2002.
[22] B. Waxman, "Routing of multipoint connections," *IEEE J. Select. Areas Commun.*, vol. 6, pp. 1617–1622, Dec. 1988.
[23] A. L. Barabasi and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, pp. 509–512, Oct. 1999.
[24] Z. Xiang, Z. Zhang, Q. Zhang, and W. Zhu, "P2P-based multimedia distribution service," *IEEE Trans. Multimedia*, to be published.
[25] S. Saroiu, P. K. Gummadi, and S. D. Gribble, "A measurement study of peer-to-peer file sharing systems," in *Proc. Multimedia Computing and Networking (MMCN)*, 2002.
[26] Y.-H. Chu, S. G. Rao, and H. Zhang, "A case for end system multicast," presented at the ACM SIGMETRICS, Santa Clara, CA, June 2000.
[27] B. Zhang, S. Jamin, and L. Zhang, "Host multicast: A framework for delivering multicast to end users," in *Proc. IEEE INFOCOM*, vol. 3, June 2002, pp. 1366–1375.

[28] B. Krishnamurthy and J. Wang, "On network—Aware clustering of web clients," presented at the ACM SIGCOMM'2000, Stockholm, Sweden, Aug. 2000.

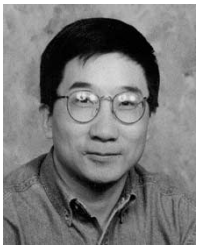
[29] ——, "Topology modeling via cluster graphs," presented at the ACM SIGCOMM IMW'2001, San Francisco, CA, Nov. 2001.

Xin Yan Zhang (S'03) received the B.S. degree in computer science from Tsinghua University, Beijing, China, in 2001. He is currently working toward the M.Phil. degree in the Department of Information Engineering, The Chinese University of Hong Kong.

His research interests include modeling, analysis and measurement on Internet, especially overlay network.

Qian Zhang (M'00) received the B.S., M.S., and Ph.D. degrees from Wuhan University, China, in 1994, 1996, and 1999, respectively, all in computer science.

She joined Microsoft Research, Asia, Beijing, China, in July 1999, as an Associate Researcher in the Internet Media Group and now is a Researcher, project lead of the Wireless and Networking Group. She has published over 80 refereed papers in international leading journals and key conferences in the areas of wireless/Internet multimedia networking, wireless communications and networking, and overlay networking. She is the inventor of more than a dozen pending patents. Her current research interest includes multimedia delivery over wireless, Internet, next-generation wireless networks, P2P network/ad hoc network. Currently, she is participating in many activities in the IETF Robust Header Compression (ROHC) WG group and is the principal contributor of the IETF ROHC WG draft on TCP/IP header compression.



Zhensheng Zhang (M'90–SM'01) received the Ph.D. degree in electrical engineering from the University of California, Los Angeles.

He is currently with San Diego Research Center, San Diego, CA. He visited Microsoft Research Asia, Beijing, China, in the summer of 2002, and worked at Sorrento Networks, Department of System Architecture, San Diego, CA, for two years, responsible for designing the next-generation optical metro networks using the GMPLS control framework. Prior to joining Sorrento Networks, he was with Bell Laboratories, Lucent Technologies, Murray Hill, NJ, focusing on research and development in ATM/SONET infrastructure and IP over WDM. He has published more than 100 papers in the IEEE TRANSACTIONS ON NETWORKING, the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, the IEEE TRANSACTIONS ON COMMUNICATIONS, and key IEEE conferences. His research interests include wireless networks, IP over WDM networks, GMPLS, and peer-to-peer networks.

Dr. Zhang was a Guest Editor for the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS for the Special Issue on Overlay Networks, 2003, and for the Special Issue on IP Over DWDM Networks, October 2000, and the *Journal of Wireless Networks* issue on Multimedia Wireless Networks, August 1996. Currently, he is Editor for the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS and on the Editorial Board of *Optical Networks Magazine*. He also served as the Chair of Technical Program Committee for OptiComm 2003 Conference, Dallas, TX, 2003. He is a recipient of the 1988 Phi Beta Kappa Scholarship Award.

Gang Song received the B.E. degree in computer science from Tsinghua University, Beijing, China, where he is currently working toward the M.S. degree in computer science and technology.

His research interest includes peer-to-peer and ad hoc networks, multimedia, and human-computer interaction.

Wenwu Zhu (S'92-M'97-SM'01) received the B.E. and M.E. degrees from the National University of Science and Technology, China, in 1985 and 1988, respectively, the M.S. degree from Illinois Institute of Technology, Chicago, and the Ph.D. degree from Polytechnic University, Brooklyn, NY, in 1993 and 1996, respectively, all in electrical engineering. From August 1988 to December 1990, he was with the Graduate School, University of Science and Technology of China (USTC), and Chinese Academy of Sciences (the Institute of Electronics), Beijing, China.

He joined Microsoft Research, Beijing, China, in 1999 as a Researcher in the Internet Media Group. He is currently a Research Manager of the Wireless and Networking Group. Prior to his current post, he was with Bell Labs., Lucent Technologies, Whippany, Holmdel, and Murray Hill, NJ, as a Member of Technical Staff from 1996 to 1999. While he was with Bell Labs, he performed research and development in the areas of Internet video, video conferencing, and video streaming over IP networks. He has published over 150 refereed papers in various IEEE journals and conferences in the areas of wireless/Internet video delivery, wireless/Internet multimedia communications and networking, and has contributed to the IETF ROHC WG draft on robust TCP/IP header compression over wireless links. He is the inventor of more than 12 pending patents. His current research interest is in the area of wireless/Internet multimedia delivery and multimedia networking.

Dr. Zhu has served as Guest Editor for the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY for the Special Issues on Streaming Video and Wireless Video, respectively. He also serves as Guest Editor for the IEEE JOURNAL SELECTED AREAS IN COMMUNICATIONS for the Special Issue on Advanced Mobility Management and QoS Protocols for Wireless Internet. Currently, he is serving as Guest Editor for the Special Issue on Advanced Video Coding and Delivery in PROCEEDINGS OF THE IEEE. He received the Best Paper Award in the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY in 2001. He is a Member of Eta Kappa Nu, the Visual Signal Processing and Communication Technical Committee, and the Multimedia System and Application Technical Committee of the IEEE Circuits and Systems Society. He is also a Member of the Multimedia Communication Technical Committee of the IEEE Communications Society.