GeoCast: An Efficient Overlay System for Multicast Applications

Yuehua Wang[1,2], Ling Liu[1] Calton Pu[1] Gong Zhang[1]

[1] College of Computing, Georgia Institute of Technology, USA

[2] State Key Lab of Virtual Reality Tech. and Systems, School of CSE, Beihang University,P.R.China

Email: {yuehua,lingliu,calton.pu,gzhang3}@cc.gatech.edu

Abstract

In this paper, we present GeoCast, a geographical location aware overlay network framework designed for providing efficient group communication services. GeoCast can be seen as an extension to the CAN network in the term of topology management and routing protocol. Geocast design has three important properties that attractive to group communication applications. First, it uses geographical mapping of nodes to regions to take advantage of the similarity between physical and network proximity. Second, a shortcut enabled geo-distance routing protocol is employed in GeoCast, which is more resilient than Chord-like or Pastry-like overlay networks due to the availability of multiple independent routing paths. Third and most importantly, a novel routing table management scheme is designed to allow the applications based on that have ability to manage their maintenance overhead in terms of network resource constrains.

I. Introduction

The rapid growth of wireless communication technology and increasing popularity of handheld devices are continuously escalating multi-party group communication applications, such as multiplayer online games, proximity based advertising, real-time conference, IPTV, and instant messaging. At the same time, the success of file sharing applications and Skype, has made decentralized overlay networks an attractive alternative computing paradigm for distributed interactive applications andwide area multicast applications. A number of research projects [1] [2] [3] have engaged in the design of peer to peer overlay networks for supporting group communication applications.

By investigating the existing studies in overlay multicast, we reach three important observations. First, for distributed wide-area group communication applications supported by overlay networks, the properties of overlay networks such as the communication efficiency, system scalability, and fault resilience largely dominate their performance. Second, the efficiency of overlay multicast, especially the efficiency of the multicast tree constructed for each communication group heavily depends on the efficiency of the underlying P2P routing protocol provided by each specific overlay network. Third, most of the overlay network topology does not well conform to the underlying IP network topology and the application and management messages are exchanged using the unicast links among end-hosts. As a result, communication messages may have to traverse the IP network from one end to another multiple times to reach its destination. This is particularly true when group communication peers are widely distributed across the Internet. For instance, Scribe [1] is a peer to peer (P2P) multicast system that builds multicast trees for efficient information dissemination to large subscriber groups by taking advantage of the peer to peer routing protocol of Pastry. Similarly, both PeerCast [2] and the proposal by [3] design and develop efficient multicast services based on Chord [4]. PeerCast enhances the Chord routing protocol by encoding network proximity information into the identifier of each peer such that physical network neighbors share similar identifier prefixes and are more likely to be clustered closer in the identifier space. The proposal in [3] enhances the performance of Chord by taken into account of the heterogeneous capacities of end-hosts.

In this paper, we describe GeoCast, a scalable and decentralized geographical overlay multicast system framework, GeoCast can be seen as an extension to the CAN network in terms of overlay topology management and routing scheme. Geocast design has a number of important properties that attractive to group communication applications. First, the geographical proximity is taken into account in the procedure of the overlay network construction, which is well known that is critical for many location-based applications, such as location based advertisement and entertainment, on-demand IPTV delivery, and online gaming applications. Second, it employs a shortcut enabled geo-distance routing protocol to speed up the routing procedure among nodes in the overlay network. Third and most importantly, a novel routing table management scheme is designed to allow the applications based on that have ability to control their maintenance overhead in terms of network resource constrains. It improves CAN-like multicast overlay system by organizing the elements in the routing table in a well defined way.

The rest of the paper is organized as follows. First, we discuss the related work in section 2 and then describe the structure of GeoCast system in section 3. Specifically, the design of routing table and maintenance scheme is described and the shortcut-enable routing scheme is studied in detail. We summarize the contribution of this paper in Section 4.

II. Basic GeoCast system Overview

GeoCast is a geographical overlay system built on top of GeoGrid[5]. It extends GeoGrid by developing multicast tree algorithms and fast utility based protocol to support end-to-end multicast services.GeoCast consists of a network of end system nodes interconnected through the GeoCast topology and routing protocol. Nodes are equipped with a GeoCast middleware composed of two-tier substrates: overlay network management and end system node multicast management.

A. Overlay network management

This substrate is the lower tier substrate for overlay membership management, lookups, and communication. It consists of membership protocol and routing lookup protocol.

1) Membership protocol: GeoCast system uses the membership protocol to organize widely distributed end system nodes into a overlay network that carries the multicast service. Each node is represented as an point in a 2D geographical coordinate space called G, which bears a one-to-one mapping to the physical coordinate system. Similar to CAN [6], at any point in time, the network of N nodes will dynamically partition the entire GeoCast coordinate space into N disjoint rectangles such that each node ”owns” a rectangle region.

Each end system node is represented by a tuple of five attributes, denoted by $E_i :< x, y, R,$ $IP\_Port, property >$, where (x, y) is the unique identifier of end system node E_i. *R* is the region that node E_i manages. It is described as a quadruple: (x, y, w, h), where (x, y) represents the coordinates of top left vertex of region and (w, h) refers to the width and height of R. Let N be the number of end system nodes in GeoCast. It subjects to (i) $\sum_{i=1}^{N} E_i.R.w * E_i.R.h = G.w * G.h$ (ii) $E_i.R \bigcap E_j.R \in \{\o, d(u, v), \overline{d_1(u', v')d_2(u'', v'')}\}$ for $i \neq j$, where $d(u, v)$ denotes a point that belongs to both two regions $E_i.R$ and $E_j.R$ and (u, v) is its coordinates. $\overline{d_1(u', v')d_2(u'', v'')}$ represents a line segment that is the intersection of those regions, where (u', v') and (u'', v'') refer to coordinates of initial point and end point of such line segment respectively. In our case, two regions $E_i.R$ and $E_j.R$ are considered immediate neighbors iff their intersection is a line segment $\overline{d_1(u', v')d_2(u'', v'')}$. IP_Port is node E_i's *IP* address and port used to communicate with other network nodes. *Property* refers to a set of specific attributes of E_i, which are related to the specific application. For different applications, it may have different meanings. It may represent the available storage space for file sharing services, the available bandwidth for streaming applications and any combination of those attributes. In GeoCast, we record node's bandwidth information in *property* field, which quantifies the amount of network resources that node E_i is willing to dedicate for serving other nodes. To keep overlay connectivity among nodes in system, each node maintains a list, say routinglist, used to record the related information about other nodes such as their unique identifier, *IP_Port*, and *property*. It helps the node maintain a partial view of other overlay nodes. Similar to GeoGrid, GeoCast is constructed incrementally. As a new node joins the system, it first contacts a bootstrapping server [5] to obtain a list of existing nodes in GeoCast and initiates a joining request with its own geographical coordinates by contacting an entry node selected randomly from this list. The joining request is then routed to the region that covers the coordinates of the new node. After identifying the specific region to which the new node belongs, the owner node makes a $unit - capacity$comparison with all of its neighbors and selects the node with least unit-capacity as the split node which partitions its region in half and assigns one half to the new node. $Unit\_capacity$ represents the workload capacity per region unit, defined by:$unit\_capacity_i = \frac{E_i.property}{E_i.R.w \times E_i.R.h}$, where $E_i.R.w$ and $E_i.R.h$ represent the width and height of region R owned by E_i.

B. End system node multicast management

This substrate is the higher layer of GeoCast middleware, responsible for multicast event handling, subscription management, multicast payload delivery, and group membership management. It is built on top of the overlay network management substrate and uses it to carry out management functions. In GeoCast, there are four distinct operations designed to complete multicast service establishment and maintenance:

Publishing the Multicast Service : multicast sources join the GeoCast overlay as peers. Each multicast service is associated with two identifiers: service identifier and group identifier. The service identifier will be used to advertise and publish meta-information about the service, whereas the group identifier would be used by peers to subscribe to and unsubscribe from the multicast service, which is typically the geographical coordinates of the publisher node.

Multicast Tree-based Subscription Management : The newly subscriber node (denoted by E_j) issues a subscription request with the group identifier of the multicast service. This subscription request is treated exactly like a lookup request on the group identifier, and is forwarded towards the multicast source through a series of intermediate nodes. Eventually, the request reaches the source node. However, in many cases it is not necessary to forward the request to the source node. If one of the intermediate nodes has already subscribed to the multicast service requested by E_j (i.e., the node is already in the multicast tree), the forwarding of the multicast subscription request is terminated. Instead, the intermediate node that is already in the multicast tree adds E_j as a new leaf to the multicast tree and starts forwarding the messages of the multicast services to E_j. Thus, we see that each subscription request adds one or more edges into the multicast tree. Conversely, the multicast tree is pruned through analogous operations when nodes *unsubscribe* from the respective multicast service [2]. Since node failures without explicit notification are very similar to unsubscribe in terms of the action that needs to be taken[5], we treat them as the same cases in our system.

Dissemination of Multicast Payload : the source of a multicast service uses the corresponding multicast tree for delivering the multicast data to all the subscribers. It injects the data at the root of the multicast tree, which then gets disseminated through the tree and reaches all the subscribers.

Multicast Group Management : as it is impractical to rely on a single node to keep track of group membership, the burden of group maintenance is shared jointly by all nodes in the group. Every group node maintains a list of multicast nodes in its group based on its local knowledge. Considering propagation of heartbeat message among group nodes could potentially be quite expensive, the group itself manages its members in absence of introducing explicit handshake mechanism, by piggybacking such information either in data message transmitted among nodes or in handshake message used by shortcut maintenance.

III. Shortcut enabled Routing scheme

It is well known that Chord achieves $O(\log_2 N)$ by forwarding a routing message according to the finger table entries of each node can reduce the search space by half at each forwarding step. In Pastry, the search space is further reduced to $1/2^d$ because each node of Pastry maintains even more routing information than Chord.

Inspired by the above analysis, we use a technique called *Routing shortcut*. The main idea is to maintain more routing information such as shortcuts pointing to other regions at each node by making use of the split history of the region managed by nodes.As new nodes continuously join the system, the nodes in the system keep splitting their regions and assign them to those nodes so that they can become one part of the system.

In this section, we first give a description of Routing shortcut, specifying how nodes' routinglists are constructed and maintained in overlay network merely based on their partial knowledge about network. Then we introduce the shortcut enabled routing scheme used in this paper.

A. Routing shortcut

1) Routinglist construction: In our scheme, instead of discarding the nodes that are no longer immediate neighbors of split node, they are preserved and used as the shortcut nodes in forwarding routing requests. For a node of identifier E_i, both neighbor nodes and shortcut nodes are organized into a list $routinglist = \{S_0, S_1, \ldots, S_i, \ldots, S_Q\}$. S_i is defined as a n subset of routinglist. It contains both node E_i's and entry nodes' responsible region in a geographical enclosing zone EZ_i with the $1/2^i$ size of the geographical plane. It subjects to the constraint that the edges of the zone are parallel to that of node E_i's region, which is represented by $EZ_i :< x, y, w, h >$, where (x, y) represents the coordinates of top-left vertex of enclosing zone and (w, h) refers to the width and height of EZ_i. The nodes in subset S_i may or may not own a region with $1/2^{i+1}$ size of plane G. In our case, those nodes are viewed as representatives of enclosing zone. If there is a message aimed to reach a node E'' contained in enclosing zone EZ_k, it is more likely for entries in subset S_i to be selected as next hop to forward the message to the destination.

As mentioned before, GeoCast starts from a node E holding the entire plane. With nodes arrival and departure, the plane is partitioned by nodes into a set of rectangle regions with different size. Nodes continuously add the nodes that are adjacent to them into their routinglists. There is no shortcut node kept in node's routinglist until it holds following condition: $log_2 \frac{G.w*G.h}{R.w*R.h} - 1 > 0$. Each time the split node who satisfies such rule creates a new subset, adds its siblings into such subset, and attaches it to the end of its routinglist. The remaining nodes in routinglist are re-assigned into the other existing subsets except the new one in terms of theirs coordinates. Correspondingly, newly joined node initializes its routinglist by learning the local knowledge of

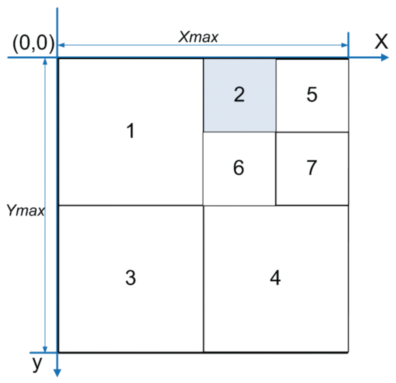

Fig. 1: An GeoCast overlay network

existing node about the network. Concretely, it inherits the list from the split node and use the contents of the entries of the list to built its own routinglist. On receiving the notification of the new node, the neighbors of the split node updates their routinglist in the same manner to accommodate the change of topology.

Initially, there are two different kinds of nodes in routinglist: neighbor node and shortcut node that is the node used to be neighbor of current node. Unlike neighbor-based routing, routing shortcut keeps those old neighbors in routinglist, which are used as shortcuts to speed up the message forwarding. In routing shortcut, the routinglist is organized by borrowing the concept of enclosing zone. It requires each entry E' in subset S_i is in the enclosing zone EZ_i but not in the enclosing zone EZ_{i+1}, i.e. for any $0 \leq i \leq Q$ and $E' \in S_i$, $(E'.x, E'.y) \in EZ_i \wedge (E'.x, E'.y) \notin EZ_{i+1}$, where $Q = \log_2 \frac{G.w*G.h}{E_i.R.w*E_i.R.h}$. EZ_i is the bounding box with $1/2^i$ size of the geographical plane within which both node E_i and node E' lie. Initially, when $i = 0$, EZ_i is identical to the entire geographical plane G that is the largest enclosing zone of node E_i. For $i \geq 1$, we have:

$$EZ_i \quad : \quad < EZ_{i-1}.x + (\theta - 1) * \frac{EZ_{i-1}.w^{\delta}}{2}, EZ_{i-1}.y + \frac{\theta * EZ_{i-1}.h^{\eta}}{2}, \frac{EZ_{i-1}.w}{\theta + (-1)^{(\theta+1)}}, \frac{EZ_{i-1}.h}{\theta} >$$

$$\delta = \lceil \frac{EZ_{i-1}.x + EZ_{i-1}.w/2 - E.x}{EZ_{i-1}.x + EZ_{i-1}.w/2} \rceil$$

$$\eta = \lceil \frac{EZ_{i-1}.y + EZ_{i-1}.h/2 - E.y}{EZ_{i-1}.y + EZ_{i-1}.h/2} \rceil$$

$$\theta = \frac{EZ_{i-1}.h}{EZ_{i-1}.w}$$

where parameter θ is used to adjust the shape of enclosing zone according to the location of node E_i. Given that, we also can get the relationships between $EZ_{i-1}$and EZ_i: $\forall\ 1 \le i \le Q$,

$$i) \quad EZ_{i-1} \supset EZ_i$$
$$ii) \quad EZ_{i-1}.w * EZ_{i-1}.h = 2 * EZ_i.w * EZ_i.h$$

For simplicity, we refer to the value of $EZ_i.w{*}EZ_i.h$ as the size of enclosing zone EZ_i(denoted by $EZ_i.size$) in the following discussions.

Figure 1 provides a snapshot of GeoCast overlay network with 7 end system nodes. By following the joining order: $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 6 \rightarrow 7$, the entire geographical plane G is partitioned as shown in Figure 1. Node 2,5,6,7 are mapped into the same zone with a quarter size of coordinate space due to the proximity of unique identifer of those nodes. For node 2, it has $neighborlist = \{1, 5, 6\}$ and $routinglist = \{\{1, 3\}, \{4\}, \{5\}, \{6\}\}$. In this case, two subsets $\{1, 3\}$ and $\{4\}$ are created and added into routinglist after the arrival of node 5 and node 6 respectively, which subjects to:

$$i) \quad node\ 1, node3 \in EZ_0 :< 0, 0, Xmax, Ymax >$$
$$ii) \quad node\ 1, node3 \notin EZ_1 :< Ymax/2, 0, Xmax/2, Ymax >$$
$$iii) \quad node\ 4 \in EZ_1 :< Xmax/2, 0, Xmax/2, Ymax >$$
$$iv) \quad node\ 4 \notin EZ_2 :< Xmax/2, 0, Xmax/2, Ymax/2 >$$

Different nodes may have their routinglist with different size. The exact length of the routinglist for a node E_i is decided by the relative size of the region R owned by E_i. When the region R has $1/2^Q$ size of the geographical plane, where $Q = \log_2 \frac{G.w*G.h}{E_i.R.w*E_i.R.h}$, the length of the $routinglist_i$ is Q, which allows to cover the entire geographical plane by its entries in routinglist of E_i according to the following equation $\sum_{i=1}^{Q} 1/2^i + 1/2^Q = 1$.

Proof of Theorem: $\sum_{i=1}^{Q} 1/2^i + 1/2^Q = 1$

Proof: Initial step: When Q = 1, we know that there are at least 2 nodes existing in the current system in terms of condition mentioned above. The node E with half of entire plane has its neighbor in its subset S_0. Thus, we have $\sum_{i=1}^{Q} 1/2^i + 1/2^Q = 1/2 + 1/2 = 1$. It indicts the statement is true for Q = 1. Induction hypothesis: Suppose the statement holds for Q=k. Induction step: Then we would like to show that it also holds when Q increase to k+1. This

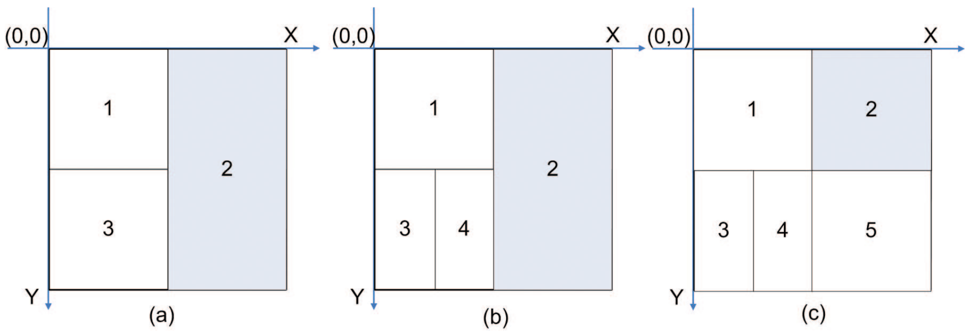

Fig. 2: An GeoCast overlay network

can be proved by the fact below. Let E' be a new node who wants to join in the plane and be mapped to node E in terms of its coordinates specified by request message. On receiving such message, the owned node E splits its region into two parts with same size and assign one part to the new node. Given the definition of enclosing zone, it is easy to know that both node E and E' are included in EZ_k that is the enclosing zone with $1/2^k$. At this moment, node E owns a region with $1/2^{k+1}$ size of G after k+1 times splitting. Thus, $\sum_{i=1}^{Q} 1/2^i + 1/2^Q = \sum^{k} +1_{i=1} 1/2^i + 1/2^{k+1} = \sum_{i=1}^{k} 1/2^i + 1/2^{k+1} + 1/2^{k+1} = \sum_{i=1}^{k} 1/2^i + 1/2^k = 1$. □ Exactly, Theorem is from the fact *For any* $0 \leq i \leq Q$ *and* $S_i \in routinglist$, *the number of nodes contained in* S_i *is greater than or equal to 1.*

Proof : we prove this by contradiction. Suppose that there is a subset S_j, such that $S_j = \{\emptyset\}, 0 \leq j \leq Q$. Then clearly when $j < Q$, the total number of nodes enclosed in EZ_j equals to that of EZ_{j+1}, which is derived from the feature of enclosing zone EZ_j that $EZ_j \supset EZ_{j+1}$. However, recall that each subset S_i is established and added into routinglist when the node's responsible region R is partitioned. And the sibling node is correspondingly included into the subset, which must be belonged to EZ_i but not EZ_{i+1} when $0 \leq i < Q$. Even if $i = Q$, the number of nodes in S_Q must be large than 0 in terms of the construction of subset in routinglist. Therefore the suppose is a contradiction. □

Figure 2(a) provides a snapshot of GeoCast overlay network with 3 end system nodes. By following the joining order:$1 \rightarrow 2 \rightarrow 3$, the entire geographical plane G is partitioned as shown in Figure 2. At this moment, there are no shortcut nodes including in the nodes' routinglist. For node 2, it has: $routinglist = \{\{1, 3\}\}$, where node 1 and 3 are its neighbors, included in EZ_0 that is identical to the entire plane G. After node 4 arrives, node 3 becomes an old neighbor of node 2 and is recorded as a shortcut node in routinglist at node 2. Similarly, node 4 become another shortcut node after node 5 joining the system, as shown in Figure 2(b)(c). And routinglist changes to $routinglist = \{\{1, 3, 4\}, \{5\}\}$ In this way, each node in the system keep building up its routinglist in terms of the changes of topology of network. It ensures that it happens only if the change of neighborhood is detected.

2) Routinglist management: In GeoCast, there are two sets of action designed to complete routinglist maintenance: update and replacement.

Update: Nodes probe their routinglist nodes for information updating in two distinct ways. Periodically, heartbeat messages containing nodes' IP address, IP_port and owned region R are exchanged among immediate neighbors. On receipt of a handshake message, nodes update their record immediately and initial the response messages including its current state to answer that message. While shortcut nodes are maintained in a lazy manner. The handshake messages are issued only when there is no message transmission among those nodes for a long time slot T_S of 2*T_N that is heartbeat interval of neighborhood updating. In our system, message transmission functions as an implicit handshake message to indicate nodes' status. A prolonged absence of handshake response indicates a failure or departure. Since the active replacement scheme works for both node departure and failure cases, we use the term failure to refer to both scenarios.

Replacement: Since the shortcuts of a node are inherited from the node that splits its region, it is likely to impose a heavy load on the nodes that participate in the system earlier. The motive of such adaption is to avoid those nodes being overloaded especially when they do not have enough capacity. This adaptation is initiated when a shortcut node is chosen to be replaced or it encounters a failure captured by shortcut probing. Periodically, each node E in GeoCast randomly chooses a shortcut node sc_i from its shortcut list for adaption. If it holds the relationship of $sc_i.unit\_capacity < Unif(0, 1) \times E.unit\_capacity$, a replacement node is chosen by random walk[7]. Unif(0,1) is a random value assigned by following uniform distribution. The random walk starts form the node sc_i chosen to be replaced. If this node is no longer in network, its coordinate will be used to locate the new owner node which takes over its region. sc_i generates a query tagged with its $unit\_capacity$ and sends it out to one of its list nodes at random. When the query arrives at the node, it checks if its $unit\_capacity$ is bigger than $sc_i.unit\_capacity$. If does, the walk stops and a response message with such node's information is sent back to node E for replacing. Otherwise, the query is issued again in the similar manner. Such design tends to keep the shortcuts that have more capacity to replace the weaker ones with high provability.

To maintain the efficiency of routing shortcut, we let each node periodically establish new relationships with nodes in the system. To do this, it first randomly selects a set of entries in its routinglist to see if they are still suitable to be served as shortcut nodes based on their current state. If shortcut node is overloaded or moved out of the system, an new shortcut node with high property value is chosen to replace such node through using random walk [8]. We omit the details due to space constraints. This operation serves two purposes: first, it helps each node maintain a relatively stable number of entries in the presence of node departures; second, it helps each node explore entries of high property.

B. *Shortcut-enabled routing*

In GeoCast, we use *shortcut-enabled Routing* to deliver messages for end system nodes. It uses geographical distance $Gdist$ as routing metric to discover routing path to the given destination that specifies in service request. $Gdist$ is the shortest distance between two end system nodes on space G, defined by $Gdist_{i \to j} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$, where E_i and E_j are end system nodes in system. (x_i, y_i) and (x_j, y_j) are the unique identifer of node E_i and E_j respectively. Destination may refer to a point or a region in space *G*. An example of such request is ”send me the current traffic state of I-85 near downtown”. In GeoCast, we tag each request with geographical coordinates *D(x, y)* which represents the spatial query point of the request. When the end system node that covers *D(x, y)* receives this request, it sends the requested information related to its region back to the request sender.

When a node p wants to route a message with the given destination coordinates, it first checks if the coordinates are contained by the region it owns and if not, it looks up routing nodes in its routinglist and choose the next routing hop node whose geographical distance to the destination coordinates is the shortest one among all candidate routing nodes in the list. If the destination coordinates are not contained by the region owned by the next routing node, this routing process repeats until the message reaches destination. Under the routing algorithm shown above,the average routing length in terms of hop count can be estimated. It ensures that any node in system can be reached at most $O(log_2 N)$ with high probability.

In GeoCast, each node has an average of O(2d) immediate neighbors and O(log N) nodes maintained in its routinglist, where d is the dimensions of coordinate space and N is the number of nodes currently in the system. Unlike CAN, it ensures that any node in the system can be

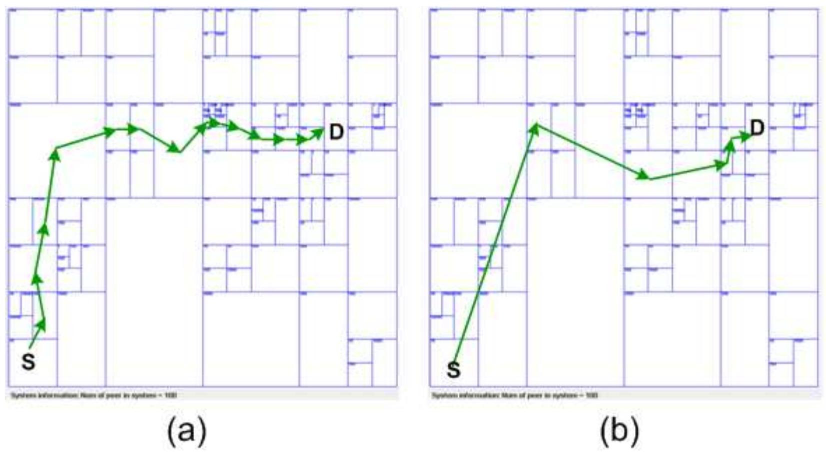

Fig. 3: Examples to illustrate neighbor based routing and Geo-distance routing

reached in less than O(logN) hops, achieving similar performance as to Pastry [1], Tapestry [9], Chord [4] and Expressway [10]. It benefits from the introduction of routing shortcut. It not only reduces the routing latency experienced by messages in terms of hop counts, but also improves the utilization of network resource. Given source node S and destination node D in a system of 100-node, figure 3 depicts the message delivery routes formed by using different schemes. Comparing to neighbor-based routing, the shortcut enabled geo-distance routing (called Geo-distance routing for short) in basic GeoCast exhibits a better performance in terms of hop counts. It only needs 5 hops to reach the destination while neighbor-based routing needs 3 times as many as that of Geo-distance routing, as shown in figure 3(a)(b). This advantage becomes more pronounced when the size of system is larger.

IV. Conclusion

In this paper, we have presented GeoCast, a middleware architecture for supporting multicast group communication applications in a geographical overlay network. Our approach has two unique features. First, one novel routinglist construction and maintenance strategy on the concept of enclosing zone is dedicatedly designed to allow the applications based on that have ability to manage their maintenance overhead in terms of network resource constrains. Second, we employ shortcut-enabled routing scheme to improve CAN-like neighbor-based routing scheme by introducing routing shortcut in the message forwarding path selection.

References

[1] Castro, M., Druschel, P., Kermarrec, A., Rowstron, A.: Scribe: a large-scale and decentralized application-level multicast infrastructure. IEEE Journal on Selected Areas in Communications **20**(8) (2002) 1489–1499

[2] Zhang, J., Liu, L., Pu, C., Ammar, M.: Reliable peer-to-peer end system multicasting through replication. In: International Conference on Peer-to-Peer Computing (P2P 2004). (2004)

[3] Zhang, Z., Chen, S., Ling, Y., Chow, R.: Resilient Capacity-Aware Multicast Based on Overlay Networks. In: International Conference on distributed computing systems. Volume 25. (2005) 565

[4] Stoica, I., Morris, R., Liben-Nowell, D., Karger, D., Kaashoek, M., Dabek, F., Balakrishnan, H.: Chord: a scalable peer-to-peer lookup protocol for internet applications. IEEE/ACM Transactions on networking **11**(1) (2003) 17–32

[5] Zhang, J., Zhang, G., Liu, L.: GeoGrid: A Scalable Location Service Network. In: Proceedings of IEEE International Conference on Distributed Computing Systems. (2007)

[6] Ratnasamy, S., Francis, P., Handley, M., Karp, R., Schenker, S.: A scalable content-addressable network. In: Proceedings of SIGCOMM Conference. (2001) 161–172

[7] Yamamoto, H., Maruta, D., Oie, Y.: Replication methods for load balancing on distributed storages in P2P networks. (In: SAINT 2005) 264–271

[8] Gkantsidis, C., Mihail, M., Saberi, A.: Random walks in peer-to-peer networks. In: INFOCOM 2004. Volume 1. (2004)

[9] Zhao, B., Kubiatowicz, J., Joseph, A.: Tapestry: An Infrastructure for Fault-tolerant Wide-area Location and Routing. Computer **74** (2001)

[10] Xu, Z., Zhang, Z.: Building low-maintenance expressways for p2p systems. Hewlett-Packard Labs, Tech. Rep. HPL-2002-41 (2002)