GeoGrid: A Scalable Location Service Network

Jianjun Zhang,Gong Zhang,Ling Liu
College of Computing, Georgia Institute of Technology
{zhangjj,gzhang3,lingliu}@cc.gatech.edu

Abstract

This paper presents GeoGrid, a geographical location service overlay network system, for providing scalable location-based services to a large and growing number of mobile users. GeoGrid is designed as a decentralized and geographical location aware overlay network and provides system-level facilities and optimizations for balancing load in the presence of node heterogeneity, dynamically moving hot-spots (location queries), and unpredictable rate of node join, departure and failure. GeoGrid uses geographical mapping of nodes to regions and geographical proximity based routing to take advantage of the similarity between physical and network proximity. Furthermore, GeoGrid exploits multiple opportunities for dynamic workload adaptation in the presence of static hotspot queries and moving hotspot queries. Its dynamic load balancing algorithms can efficiently utilize the heterogeneous capacities of end systems and balance both the location query workload and the routing workload.Our initial prototype development and experimental study demonstrate that GeoGrid can effectively reduce the workload imbalance by an order of magnitude.

1 Introduction

Advances in mobile hardware and the increasing sophistication of handheld software have made many devices location-aware. While location-based services exist for specialized niche markets such as geographical information systems (GISs), they are typically constrained to a fixed set of moving objects and expensive to maintain and expand. This is in large part because the first generation of GIS systems and applications provide little or no system support for serving a large and growing number of mobile users and providing continuous delivery and dissemination of location-based information in real time.

One approach to address this problem is creating and maintaining a centrally managed geographical location service, which can be queried and updated by the mobile users on the move via the infrastructure wireless networks. The potential drawbacks of this approach are several. First, the response time may not meet the real-time requirements. Second, the access to the infrastructure communication service is expensive. Third, the centralized approach is not robust and particularly vulnerable to failures and sudden surge of hot spots. Furthermore, there is currently no business model to provide a return-on-investment for setting up and operating such large scale location-based services.

In this paper we present GeoGrid, a geographical location service overlay network, for providing scalable location-based services to a large and growing number of mobile users. By design, GeoGrid is a decentralized and yet managed overlay network and it has three distinct characteristics. First, GeoGrid is designed on top of a geographical two-dimensional coordinate space that bears one to one mapping to the physical coordinate system. The entire coordinate space is dynamically partitioned among all the nodes in the system such that every node "owns" its individual distinct region within the entire coordinate space. GeoGrid uses geographical mapping of nodes to regions and geographical proximity based routing to take advantage of the similarity between physical and network proximity. Second, GeoGrid introduces the concept of dual peer. On one hand, dual peer empowers each owner node with fail-over capability, improving fault tolerance. On the other hand, dual peer enhances the routing efficiency by reducing the number of region split operations upon the join of new nodes, thus reducing the routing path (number of hops) and per-hop latency. Third but not the least, GeoGrid employs several dynamic load balancing algorithms to efficiently utilize the heterogeneous capacities of end systems and balance both the location query workload and the routing workload. Our initial experimental study demonstrates that GeoGrid can effectively reduce the workload imbalance by an order of magnitude.

2 Basic GeoGrid System

In this section we describe the design of our geographical location-based service grid in its most basic form, and we refer to it as the Basic GeoGrid. We will describe the

27th International Conference on Distributed Computing Systems (ICDCS'07)
0-7695-2837-3/07 $20.00 © 2007 IEEE

advanced design features such as dual peer, dynamic load balancing solutions in the subsequent sections.

GeoGrid consists of a network of nodes interconnected through the GeoGrid topology and routing protocol. All nodes are represented as points in a two dimensional geographical coordinate space, which bears a one-to-one mapping to the physical coordinate system. Similar to CAN [11], at any point in time, the network of N nodes will dynamically partition the entire GeoGrid coordinate space into N disjoint rectangles such that each node "owns" a rectangular region. For example, Figure 1 shows a two dimensional geographical coordinate space partitioned among 15 GeoGrid nodes. Nodes in GeoGrid self-organize into an overlay network. A node establishes its overlay connectivity with other nodes through its immediate neighbors. A mobile user may use the GeoGrid service network by connecting her mobile devices to one of the nodes in the network, usually through wireless or wired network connection. Each node runs the GeoGrid middleware and serves as the proxy for the mobile users. Depending on the applications implemented over the GeoGrid middleware, a proxy can submit queries, process data, cache query results, and send event notifications on behave of mobile users. A proxy can be a personal computer or a server at the edge of the Internet.

The GeoGrid development is based on a number of assumptions. First, we assume that there exist information sources that can provide the geographical contents requested by the end users. In the examples given above, such information sources could be the traffic monitoring cameras, the owners of gas stations and parking lots, and the people who are willing to share their current location information. Second, we assume that the people asking for information from our service network can be from outside of the network or be inside the network. By "outside of" the network, we mean that one can be just a consumer of the information, without being a part of the network and sharing any of its resources. Finally, to simplify our design, we assume that the network nodes are not mobile. Compared to mobile devices, desktop computers usually have more access network bandwidth, more stable connections, and more storage space. The latest survey [1] shows that more than 60% of American families have at least one personal computer, and more than 54% of American families have Internet accesses. Furthermore, advances in wireless communication technologies have enabled the point-to-point TCP/IP communication between mobile devices and fixed nodes.

Specifically, the design of GeoGrid is intended to address the following three open challenges. First, can we design a scalable network topology for constructing a geographical location based service grid such that end nodes tagged with geographical information can be effectively organized into an efficient service network that is geographical proximity aware? Second, how do we design such a system that end-to-end communication between any two end-system nodes is bounded? Third, the location-based information usually shows certain spatial and temporal clustering patterns. For example, the highway system in a metropolitan area is usually heavily loaded during the rush hours. In the morning, the highways leading in town are usually crowded, while the out-town routes are heavily loaded in the afternoon. Thus the third challenge is how to support location-based queries using heterogeneous end-systems and how should the GeoGrid service network handle such workload imbalance gracefully, minimizing the possible service interruption.

2.1 GeoGrid Construction

GeoGrid network is described by the number of nodes (N) and the two dimensional coordinate space corresponding to a geographical area of interest, such as metropolitan area, state, country, one or more continents. The entire GeoGrid plane is divided among the nodes currently in the system into a set of rectangular regions, each of which represents a rectangular area in the geographical plane. A *region* r is denoted as a quadruple in the form of <*x, y, width, height*>, where (*x, y*) represents the longitude and the latitude coordinate of the southwest corner of r, and (*width, height*) represents the x-dimension and y-dimension of the region. Two regions are considered neighbors when their intersection is a line segment. We say that a point o(*x, y*) is $covered$ by a region r if and only if the following relationship holds: $(r.x < o.x \leq r.x + r.width) \wedge (r.y < o.y \leq r.y + r.height)$.

A *node* p is identified by a tuple of five attributes: <*x, y, IP , port, properties*>. (*x, y*) represents the geographical coordinate of node p. (*IP, port*) is the IP address and port number that this node uses to execute GeoGrid middleware. $properties$ represent application specific information such as $capacity$, which quantifies the amount of resources that node p is willing to dedicate for serving other nodes. For different applications, capacity of a node may have different meanings. It may represent the available storage space for file sharing services, and the available uploading network bandwidth for multimedia streaming applications. In GeoGrid, we use *c*apacity to represent the available network bandwidth of a node. Each node p maintains a list of its immediate neighbor nodes. A node q is called an immediate neighbor of p if and only if the region owned by q and the region owned by p are neighbor regions. .

GeoGrid is constructed incrementally. It starts with one node who owns the entire GeoGrid space. As a new node joins the system, it uses its own geographical coordinate, obtained by a GPS for example, to map itself to a rectangular region of the GeoGrid that corresponds to a geographical

area in which it physically resides. After identifying the region to which the new node belongs, the owner node q of the region splits this region in half, retains half, hands the other half to the new node p, and notifies its neighbor nodes of the new node p such that the new node can be added into the neighbor list of these nodes. The new node p initializes its neighbor list using the neighbor list of q.

The basic bootstrapping process for a new node p proceeds in three steps. First, node p obtains its geographical coordinate by using services like GeoLIM [5] of GPS (Global Positioning System). Second, node p obtains a list of existing nodes in GeoGrid from a bootstrapping server or a local host cache carried from its last session of activity. In the third step, node p initiates a joining request by contacting an entry node selected randomly from this list. The joining request is routed to the region that covers the coordinate of the new node in a manner similar to the routing of a query request. The owner node q of this region splits this region in half by following a certain ordering of the dimensions such as latitude dimension first and then longitude dimension, and retains one half and hands the other half and its neighbor list to the new node p. Node p will clean its neighbor list by removing the neighbor list entries pointing to regions that are not adjacent to its region. The node whose region is split notifies the peers in its neighbor list of the new node.

2.2 Routing in GeoGrid

Routing requests in GeoGrid are location queries, each consisting of a spatial query region, a filter condition, and a focal object who issued the location service request. To simplify the discussion we tag each request with a geographical coordinate (*x, y*) which represents the spatial query region of the request, denoted by (x, y, W, H). If the query region is specified in a circle with radius γ, this spatial query region can be represented as $(x, y, 2\gamma, 2\gamma)$. In addition, we assume the focal object of each request is an existing GeoGrid node. If a mobile user issues a service request, the request will be sent to its entry node on GeoGrid through the boostraping module to be discussed in the next subsection.

Intuitively, routing in a GeoGrid network works by following the straight line path through the two dimensional coordinate space from source to destination node. A routing request is forwarded initially from its source initiator node to one of its immediate neighbors, say q, which is the closest to the destination location (*x, y*). If (*x, y*) is covered by the region owned by the chosen routing node q, then the node q will be the executor node of this request. Otherwise, q starts the forwarding process again until the request reaches the region that covers (*x, y*). Figure 1 visualizes a GeoGrid system with 15 nodes. A routing request is initiated by region 13 for a point covered by region 5. The request is forwarded through regions 10, 11, 6, and 7.

Given a GeoGrid plane of N regions, routing between a pair of randomly chosen regions has the overhead of $O(2\sqrt{N})$ in terms of the number of routing hops. Two sets of messages are exchanged in a GeoGrid network. One set of messages are for the management of GeoGrid service network, and includes messages for splitting and merging region, heart-beat, request routing, load-balancing, routing table maintenance, and randomization of routing entries. The syntax of these messages is defined by the GeoGrid middleware, and the exchanging of these messages is transparent to the applications built on top of the GeoGrid service network kernel. The other set of messages are applications specific and supported by GeoGrid middleware. Though applications define the syntax of these messages, we require the messages routed using GeoGrid middleware to supply the geographical coordinates of their destination locations.

In GeoGrid, end users submit their requests with an identified rectangular geographical area $<$ *x, y, width, height* $>$. An example of such requests is "Inform me of the traffic around Exit 89 on I-85 in the next 30 minutes". Assuming this request is issued by a node p. If node p does not reside in the spatial area of the query region, the request is forwarded to one of the neighbors of p and this process repeats until the node owns the query region is reached. In each step, the messages are forwarded in a deterministic manner. In other words, the request is routed toward the region that covers the center of query region of the request, which is a point with coordinate (*x + width/2, y + height/2*). In Figure 1, given that the center of the gray rectangular area is covered by region 5, the request is forwarded first to the node that owns region 5. From there, the owner node of region 5 examines its neighbors to see if any of them intersect with the spatial area of the request and forwards the request to those that have overlapped with the query region. In our example, region 2 and region 3 will receive the subscription request.

From this example, we can observe that load balance and routing efficiency are the two critical challenges for making GeoGrid a scalable and efficient location service network for geographical information dissemination. In the basic form of GeoGrid, the size and location distribution of regions are decided by the joining timing sequence and the geographical locations of nodes (i.e., where the nodes physically reside). When the system demonstrates non-uniform distributions in terms of node locations and node capacities, the workload assignment through geographical mapping of nodes to regions will demand a scalable and effective load balancing scheme to ensure the scalability of the GeoGrid system.

Furthermore, the load balancing scheme needs to be able to dynamically adjust the distribution of workload and alleviate the unbalanced overloading in the overlay due to (i)

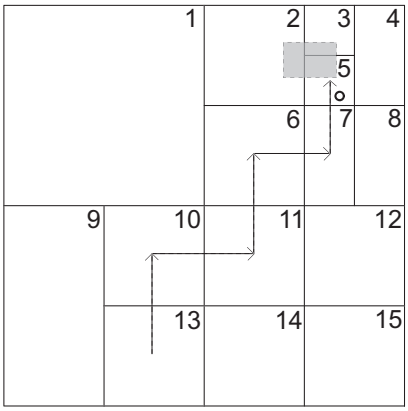

Figure 1: Basic GeoGrid service network

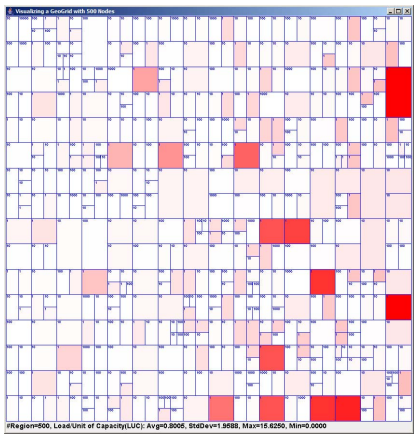

Figure 2: Region size and load distribution of GeoGrid, using random bootstrapping algorithm

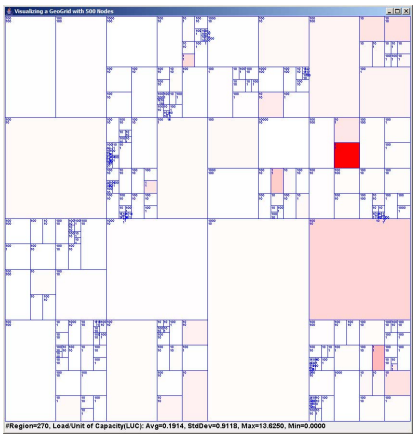

Figure 3: Region size and load distribution in the GeoGrid featured with the dual peer technique

the inherent heterogeneity in node capacity, (ii) the unbalanced concentration of nodes in some regions, and (iii) the presence of hot spot queries and the continuous movements of the hot spot queries. In GeoGrid we propose a heuristic load balance scheme that enables the workload distribution to be dynamically adjusted. Such a scheme is composed of two techniques. *Dual Peer* technique improves the overall system reliability and generally maps the region sizes to the capacities of region owner nodes. *Load Adaptation* techniques include a number of adaptation mechanisms that can dynamically adjust the node-to-region assignment among regions in geographical vicinity. We dedicate Section 2.3 and Section 2.4 to address the dynamic load adaptation problem.

2.3 Dual Peer GeoGrid

In the dual peer version of the GeoGrid, instead of using a single node as the owner to handle the requests mapped to a geographical region, we allow two nodes to share its ownership. The node with more capacity serves as the *primary owner* node, and handles all the requests and stores information in the same way as described in Section 2.1. The other node, we call it the *secondary owner* node, will serve as the backup of the primary node, holding the replication of query request and application-specific data copied from the primary node.

The basic GeoGrid needs to be modified to support the dual peer mechanism. Concretely, the following three aspects of the system design are revised accordingly.

Node Join The first three steps in which a new node p follows to join a GeoGrid network in the basic GeoGrid will remain the same. (1) The new node p obtains its geographical coordinate. (2) p uses a bootstrapping service to randomly choose an existing node as its entry point. (3) p uses the routing interface of GeoGrid to locate the region r that covers its geographical coordinate. After the new node obtains the information of the primary owner $r.primary$ of region r, it will not directly trigger the split operation on r executed by the existing primary or secondary node. Instead, it will probe the neighbor regions $r.neighbors$ of r, and will choose, from the set of regions in $r.neighbor \cup r$, a region that is not complete in terms of dual peer and the owner of which has the least available capacity. If all the regions in $r.neighbor \cup r$ are equipped with dual peer, p will choose and split the region whose primary node has the least available capacity. Between the two new regions generated by the splitting, node p will join the one whose owner has less available capacity. When node p joins a region that is half full, it will compare its capacity with the capacity of the existing owner, and will take over the role as the primary owner if the current owner has less capacity than it. The switching of primary and secondary ownerships will happen after the new node finishes copying all the objects and status information from the existing owner.

Node Departure If a region has two owner nodes, the departure of the secondary owner will cause no change to the GeoGrid system other than triggering the primary node to mark this region as "half full". The departure of the primary owner will cause the activation of the secondary owner. The new owner will inform the neighbor regions of the change and ask them to update the routing information of their primary and secondary owners.

Failure Recover The primary and secondary nodes of a region periodically synchronize their status information and

exchange heartbeat messages at a higher frequency than among the primary nodes of different regions. The failure of a node will leave its region with one or no owner. If a region is full and its primary owner node fails, the secondary owner node will take over its role, activate all the backup information, and notify the neighbors and other nodes of such a change. If the failing node is the last owner of a region, the repairing process of the basic GeoGrid network will be triggered. Otherwise, the region will be left half-full and will be filled by either a node joining later or by the load balancing adaptation scheme (to be discussed later in section 2.4).

The dual peer feature provides GeoGrid with three advantages. First, it improves the fault resilience of the GeoGrid service network. Second, it reduces the number of region split operations that may cause service interruption in the basic GeoGrid systems. The third advantage of dual peer technique is its role in improving the system load balance. A new node probes existing neighbors of the region that covers its coordinate, and joins or splits the region with the weakest primary node. Such a process will leave the regions owned by powerful nodes split fewer times and will reduce the size of the regions owned by weaker nodes. Figure 2 and Figure 3 is the visualization of a GeoGrid service network of 500 nodes. Comparing it to Figure 2, we have two observations. First, there are fewer regions and the sizes of them are distributed in less uniform manner, conforming to the capacity distribution of owner nodes. More powerful nodes now own bigger regions. Second, the selective joining process of duel peer technique renders fewer heavily loaded regions, although a few still exist.

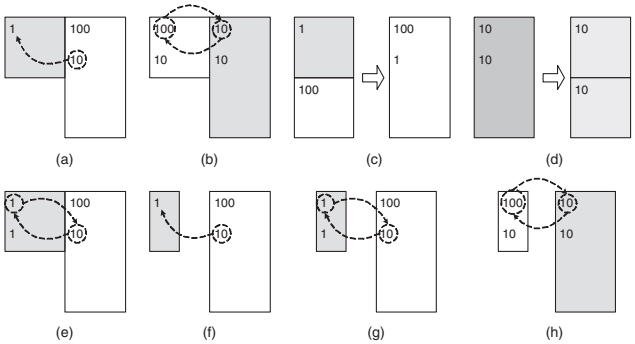

Figure 4: Load balance adaptation mechanisms of GeoGrid

2.4 Dynamic Work Load Adaptation

Recall the GeoGrid of 500 nodes visualized in Figure 3, we can still see a few heavily loaded regions (regions with darker shade), even though the total number of such regions is much smaller than that of Figure 2. Those overloaded regions all have relatively weaker primary owner nodes. Dual peer technique can balance workload distribution by selectively assigning new nodes to the most heavily loaded regions in the neighborhood of the new nodes. However, when the nodes in a region are all weak ones, the effectiveness of dual peer scheme will be less significant.

To further improve the system load balance of GeoGrid service networks, we develop a set of dynamic load adaptation mechanisms. The basic idea behind those adaptation mechanisms is to break the geographical association between an owner node and the region it owns, and dynamically adjust the node assignments in a geographical vicinity according to the workload distribution. Figure 4 illustrates eight adaptation mechanisms we use in GeoGrid. Each of them describes an adaptation scenario. The capacities of the primary and secondary owners are printed in the upper left corner of each region. There are three basic rules: (1) Local adaptations have less operation overhead than remote adaptations, and thus have higher priority. (2) Switching or moving secondary peers has less operation overhead than switching or moving primary peers. (3) Region splitting and merging are expensive operations and are thus assigned with the lowest priority.

In a GeoGrid network, each node periodically exchanges workload statistic information with its neighbors. A node starts its load balance adaptation process only when its workload index is higher than $\sqrt{2}$ times of the lowest one among its neighbors and there are no new nodes that are ready to join this region. By doing so, we can avoid the load balance adaptation process being repeatedly triggered within a geographical area in a certain time window. Whenever the load balance adaptation condition is satisfied, one of the following eight mechanisms will be used to adjust the owner node assignments and they are described in the order of increasing cost of adaptation. Due to space constraint, we omit the algorithm that describes the procedure of how these dynamic load adaptations are carried out at runtime.

(a) Steal Secondary Owner This adaptation is used when the overloaded region has no dual peer (half full). Using this adaptation, the overloaded primary owner node compares the workload index of all the neighbor regions to select a neighbor region whose secondary owner is more powerful than itself, and has the lowest workload index among all the regions satisfying the first condition. Once such a region is located, its secondary owner is "stolen" to be the primary owner of the overloaded region (see Figure 4(a).

(b) Switch Primary Owners This adaptation can be initiated by a region that is either half-full or full. Figure 4(b) gives an example. A smaller region has a primary owner that is more powerful than one of its neighbor regions, which is bigger and has a weaker primary owner. By switching the primary owners of these two regions, the bigger re-

gion now has more processing power while the smaller one has less.

(c) Merge with a Neighbor This adaptation is used when a region p and one of its neighbor region n can be merged, and the merged region has lower workload index than the average workload index of p and n. An example is given by Figure 4(c).

(d) Split a Region As illustrated in Figure 4(d), if the primary and secondary owner of an overloaded region have the same capacity, splitting this region can assign half of the workload to each of them and can reduce the workload index of the original primary owner by half.

(e) Switch Primary with Neighbors Secondary Owner When an overloaded region has a dual peer (full), it means both nodes have less capacity to handle the workload. Thus the primary owner of the region can switch its position with a secondary owner of a neighbor region, if that secondary owner has more capacity. An example of such adaptation is given by Figure 4(e).

(f) Steal Remote Secondary Owner It is possible though infrequent that a region and all its neighboring regions are overloaded. In such a case GeoGrid runs a Time to Live (TTL) guided search for the remote region whose secondary owner has more capacity than the primary owner of the overloaded region and is less loaded. As illustrated by Figure 4(f), after a remote secondary owner is discovered, the primary owner of the overloaded region will steal this remote secondary owner, and will resign to be the secondary owner.

(g)Switch Primary with Remote Secondary Owner This adaptation is for a full region – the region that has dual peer, and both primary node and secondary node have less capacity than required to handle the current workload demand. The overloaded primary owner will switch its position with the discovered remote secondary owner that is stronger than itself based on the guided search, as shown in Figure 4(g).

(h) Switch Primary with Remote Primary Owner This adaptation is for a full region and is also based on a search for discovering a candidate remote primary owner that is stronger than the primary owner of the overloaded region. The overloaded primary owner will switch its position with the discovered remote primary owner, as shown in Figure 4(h).

Note that each time a node launches its adaptation process, it will begin with the least expensive operation. Expensive ones like switching primary owner with remote primary owner are used only when all the other adaptations fail.

3 Experimental Results

We evaluate GeoGrid system by simulating a geographical region of 64 miles $\times$ 64 miles. The population of end users in this region ranges from 1×10^3 to 1.6×10^4. For each population setting, we simulated 100 randomly generated GeoGrid service networks. Each end user connects into the GeoGrid service network through a dedicated proxy node. The capacities of those proxies follow a skewed distribution based on a measurement study of Gnutella P2P network [12].

3.1 Balance Region Workload

We design a set of experiments to evaluate the effects of dual peer and load balance adaptation techniques we proposed in Section 2.3 and Section 2.4 We simulate the uneven workload distribution in the geographical area by creating a number of hot spots and randomly moving them in the simulated plane. Each hot spot is a circular area with a random initial radius between 0.1 and 10 miles. The cell at the center of a hot spot has the highest normalized workload 1 and the ones on its border have workload 0. The workloads of cells covered by the hot spot is decided by a formula $1 - d/r$, where d is the distance of a cell to the center of the hot spot and r is the radius of the hot spot. We choose circular hot spots because this choice agrees with the nature of location-based applications. To illustrate this point, let us use the queries on parking lots information as an example. Usually during a sport event like Super bowl, parking lots close to the stadium are usually fully loaded. More people will be interested in finding a parking space that is closer to the stadium. As we move from the stadium to the parking lots in the neighborhood, fewer people will be interested in querying them because parking there means longer walking to the stadium. Consequently, as the sport event creates a hot spot of queries in that area, more queries will be forwarded towards the center of the hot spot, and fewer will be forwarded to the nearby regions.

The whole simulation time line is divided into a number of epoches. At the end of each era, we force each hot spot to migrate along a randomly chosen direction and at a random step size uniformly chosen from range $(0, 2r)$. We simulated three types of GeoGrid system: basic GeoGrid system, the one featured with dual peer technique, and the one with both dual peer and load balance adaptation turned on. We measure the max, mean, and standard deviation of

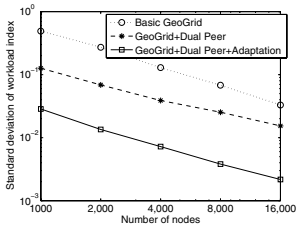

Figure 5: Standard deviation of workload index

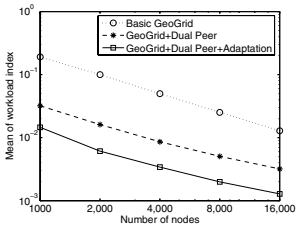

Figure 6: Mean of workload index

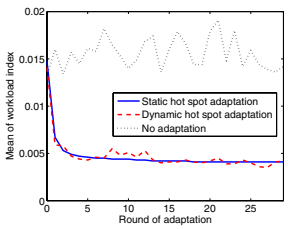

Figure 7: Convergence of the mean workload index in adaptation, plotted by round of adaptation

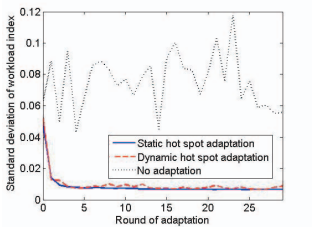

Figure 8: Convergence of the standard deviation of workload index in adaptation, plotted by round of adaptation

workload index for each version of the GeoGrid. Each simulation is repeated 100 times on different node population.

Figure 5 and Figure 6 show that both dual peer and adaptation techniques can effectively improve the load balancing in GeoGrid systems. The GeoGrid system with both features can constantly beat the basic GeoGrid system by one order of magnitude in both metrics. While dual peer technique itself can improve the workload distribution, the load balance adaptation can further improve the system performance.

3.2 Impact of Adaptation

Both dual peer and load balance adaptation techniques can improve the workload distribution of GeoGrid systems. We want to know how fast load balance adaptation can improve the workload distribution. Will the adaptation converge? To answer this question, we design a simulation to evaluate the effects of load balance adaptation on GeoGrid systems. We simulate GeoGrid systems with 2×10^3 peers. The service network is setup first using only dual peer technique. When hot spots appear, we turn on the load balance adaptation features on each node. The max, mean, and standard deviation of workload index of all the nodes are recorded at the end of each round of adaptation. We simulated a "static hot spot" adaptation scenario in which the hot spots are static and never change their size or location. The "moving hot spot" scenario is simulated by constantly moving the hot spots at a pace far faster than the pace of adaptation. Concretely, hot spots move 4 to 10 steps before a round of adaptation ends.

Figure 7 and Figure 8 plot the experiment results. The dashed line represents the measured results of the "moving hot spot" scenario while the solid line represents the result of the "static hot spot" scenario. Under both setups, the workload distribution of GeoGrid system converges in the first a few rounds of adaptations. After that, the whole system is stable enough to accommodate the constant moving hot spots without being overloaded. The dotted line represents the performance of a GeoGrid system with no load balance adaptation mechanism under "moving hot spot" scenario. Compared to the number of GeoGrid system featured with load balance adaptation mechanisms, we can see that the adaptation greatly improved the system load balance.

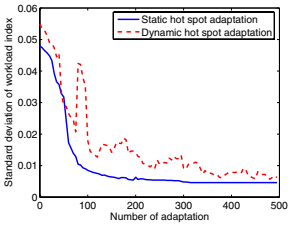

Figure 9: Convergence of the standard deviation of workload index in adaptation, plotted by number of adaptation

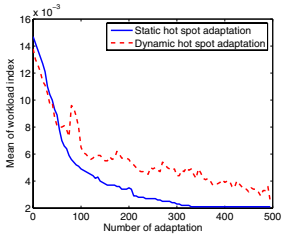

Figure 10: Convergence of the mean workload index in adaptation, plotted by number of adaptation

Figure 9 and Figure 10 plot the recorded load balance measurement results. While the lines of "static hot spot" scenario converge quickly, the ones of adaptation under "moving hot spot" scenario converges slower than in the figures plotted by the total number of adaptations. In the middle, there are a few surges on the dashed lines, which are caused by the hot spots that move to new locations in between of the first a few adaptation rounds. After a few rounds of adaptations, the whole system converges and the migrating of hot spots is handled more gracefully by GeoGrid systems.

4 Related Works

GeoGrid is inspired by research study on the d-dimensional CAN network [11, 13]. Although a number of solutions have been proposed for load balance in multidimensional CAN networks, these solutions are focused on modifying the bootstrapping process of CAN to smartly choose regions to split, which may improve the load balance of a network with more static workload distribution. However, location-based service networks that have dynamically migrating hot spots like those handled by GeoGrid,

more flexible and more responsive load balance mechanisms are needed to improve the system workload distribution. The GeoGrid dynamic load adaptation mechanisms can help achieving this goal by dynamically adjust the node distribution to accommodate the changes in workload distribution.

Research works, such as Rebeca [10], Siena [3], are focused on efficient location-based query processing using publish-subscribe (pub-sub) networks. Their solutions focus more on content-based matching of publications to subscriptions in the pub-sub network dynamically, and did not address the issue of load balance and routing efficient in terms of hop count. GeoGrid development can be used as an infrastructure for publish-subscribe applications in mobile environments, support the techniques proposed in [10, 4, 6] while adding important dynamic load balance and routing efficiency to those solutions.

Other types of networks that can support geographical information dissemination and sharing include Ad-hoc networks [9, 8], Sensor networks [2], and vehicular networks [7]. They usually use multi-hop wireless connections to implement IP unicast features among mobile or fixed nodes, and lack the fixed infrastructures like the GeoGrid service network we present in this chapter. Therefore, information dissemination in these types of networks has to rely on the unreliable wireless connections, and usually by exploiting the broadcast nature of wireless network links.

5 Conclusion

We presented the design and development of GeoGrid, a geographical location service network for scalable and efficient information delivery and dissemination to a large and growing number of mobile users. GeoGrid design is unique in three aspects. First, it promotes the use of geographical mapping of nodes to regions and geographical proximity based routing, which take advantage of the similarity between physical and network proximity to distribute the information processing and dissemination workloads according to the geographical distribution and capacity of end-systems. Second, GeoGrid utilizes the concept of dual peer to exploit multiple opportunities for dynamic workload adaptation in the presence of static hotspot queries and moving hotspot queries. Its dynamic load balancing algorithms can efficiently utilize the heterogeneous capacities of end systems and balance both the location query workload and the routing workload. Our initial prototype development and experimental study have demonstrated that GeoGrid can effectively reduce the workload imbalance by an order of magnitude.

Aknowledgement This work is partially supported by grant from NSF SEGR, NSF CSR, and an IBM faculty award.

References

[1] Computer and Internet use in the United States: 2003. http://www.census.gov/prod/2005pubs/p23-208.pdf, July 2006.

[2] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *IEEE Communication Magazine*, 40(8):102–114, 2002.

[3] A. Carzaniga, D. S. Rosenblum, and A. L. Wolf. Design and evaluation of a wide-area event notification service. *ACM Transactions on Computer Systems*, 19(3):332–383, 2001.

[4] L. Fiege, F. C. Gartner, O. Kasten, and A. Zeidler. Supporting mobility in content-based publish/subscribe middleware. In *Middleware*, 2003.

[5] B. Gueye, A. Ziviani, M. Crovella, and S. Fdida. Constraint-based geolocation of internet hosts. In *Proceedings of Internet Measurement Conference*, pages 288–293, 2004.

[6] Y. Huang and H. Garcia-Molina. Publish/subscribe in a mobile environment. In *MobiDE*, 2001.

[7] J. Luo and J.-P. Hubaux. A survey of inter-vehicle communication. Technical Report IC/2004/24, School of Computer and Communication Sciences, EPFL, 2004.

[8] C. Maihöfer. A survey on geocast routing protocols. *IEEE Communications Surveys and Tutorials*, 6(2), 2004.

[9] M. Mauve, J. Widmer, and H. Hartenstein. A survey on position-based routing in mobile ad hoc networks. *IEEE Network Magazine*, 15(6):30–39, November 2001.

[10] G. Muhl, A. Ulbrich, K. Herrmann, and T. Weis. Disseminating information to mobile clients using publish-subscribe. *IEEE Internet Computing*, 08(3):46–53, 2004.

[11] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A scalable content addressable network. In *Proceedings of SIGCOMM*. ACM, 2001.

[12] S. Saroiu, P. Gummadi, and S. Gribble. A measurement study of Peer-to-Peer file sharing systems. In *Proceedings of MMCN*, San Jose, CA, August 2002.

[13] D. Takemoto, S. Tagashira, and S. Fujita. Distributed algorithms for balanced zone partitioning in content-addressable networks. In *Proceedings of the 10th International Conference on Parallel and Distributed Systems (ICPADS'04)*, page 377, 2004.