This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the ICC 2007 proceedings.

Local Tree Based Geometric Routing

Shengpu Liu
Dept. of Computer Science and Engineering
Lehigh University
Bethlehem, USA
shl204@lehigh.edu

Liang Cheng
Dept. of Computer Science and Engineering
Lehigh University
Bethlehem, USA
cheng@cse.lehigh.edu

Abstract—**In this paper, we present a new geometric routing – *Local Tree based Greedy Routing* (*LTGR*) – for mobile ad-hoc networks. LTGR is stateless and overcomes shortcomings caused by planarization errors of previous geometric routing protocols. Local trees are constructed and their information is used to route packets bypassing void areas when the greedy geometric routing does not work. Simulation results show that LTGR outperforms GPSR (Greedy Parameter Stateless Routing) in terms of delivery ratio, routing overhead, and hop stretch. LTGR can reduce the routing overhead by 25 ~ 40% and hop stretch by 30 ~ 50% comparing to GPSR in our extensive simulation scenarios.**

Keywords-geometric routing; greedy algorithm;completeness

I. INTRODUCTION

In Mobile Ad-hoc NETworks (MANET), how to select an optimal route from the source to the destination is a critical issue. Due to node mobility and link/channel dynamics, a link that exists now between two nodes may not be subsistent in the future. Therefore, many routing protocols [1, 2, 3] have been proposed for various scenarios of MANET, which can be roughly divided into three categories: on-demand routing, table-driven routing, and hybrid routing. In on-demand routing, nodes only maintain route information when they need to send or relay packets. However, on-demand routing has longer response time than table-driven routing, and it has the scalability issue caused by the flooding of routing requests. In table-driven routing, each node always maintains up-to-date routing information to any other nodes. It would induce a heavy overhead for maintaining the routing information in highly mobile scenarios. The hybrid routing is designed to achieve a tradeoff between the characteristics of the on-demand and table-driven routing, mostly with a cost of high algorithm complexity. However, how to improve the routing scalability as well as reducing the routing overhead in MANET is still an open problem.

Geometric routing is a special type of routing approach designed for MANET where data packets are routed based on position information when node positions are known. In general, geometric routing is simple and efficient, and it improves the routing scalability because each node only needs to keep its neighbors' position information. There are common assumptions when designing geometric routing protocols. First, every node knows the position of itself. This information can be collected by using GPS devices or other means [8]. Second, every node knows its neighbors' positions that can be obtained by one hop beacon messages. Third, the source node knows the destination position. This function can be provided by some location service mechanisms [4]. As the development of positioning devices such as GPS, geometric routing is becoming more and more practical.

Most existing geometric routing protocols are based on the greedy algorithm, in which every forwarder chooses the neighbor that is the closest to the destination as the next hop. Although the greedy algorithm is simple and efficient, it fails in void areas when a node cannot find a neighbor, which is closer to the destination than itself, to forward a packet. To guarantee the packet delivery, some geometric routing algorithms, such as GPSR (Greedy Parameter Stateless Routing) [4] and GOAFR (Greedy Other Adaptive Face Routing) [5], use face routing to bypass void areas. Face routing only works in planar graphs where there is no cross link. However, failures of face routing based on planarization have been observed in some testbed [6][7] due to inconsistent radio ranges and asymmetric links.

In this paper, we propose a new geometric routing protocol for MANET called LTGR (Local Tree based Greedy Routing). LTGR uses a local tree, instead of face routing, to recover routing via bypassing void areas. LTGR only needs the same assumptions as those required by existing geometric routing protocols. It does not require assumptions of uniform radio ranges and bi-directional links that are hard to be satisfied in real-world scenarios.

LTGR has the following features. It is simple and stateless so that it is suitable for highly dynamic MANET. LTGR does not rely on planarization thus it keeps cross links in the network topology to achieve good hop stretch performance. It augments the greedy algorithm with routing history data to make informed decision in routing. Extensive simulations have been conducted to evaluate LTGR in a variety of MANET scenarios. Results show that it outperforms GPSR in terms of the packet delivery ratio, routing overhead, and hop stretch.

The paper is organized as follows. In Section II, we discuss the related work. Section III covers the basic ideas and implementation considerations of LTGR. Simulation results are described and analyzed in Section IV. We conclude this paper in Section V.

II. RELATED WORK

There exists ongoing research on geometric (position based) routing protocols [4, 5, 6, 9]. The simplest one is based on the greedy algorithm by which each node chooses its neighboring node closest to the destination as the next hop when forwarding traffic as a forwarder. However, the greedy algorithm can not pass any void area where a forwarder can not find a neighbor that is closer than itself to the destination.

In order to recover packet routing from void areas and improve the packet delivery ratio, Karp et al. propose GPSR [4] to switch from the greedy mode to a perimeter mode if a node cannot find the next hop using the greedy algorithm. In the perimeter mode, face routing [4, 10] combined with a right-hand rule is utilized to traverse the perimeter of the void area. Packets in the face routing travel along the perimeter of the faces, which are intersected by the virtual line between the source and the destination. GPSR only uses the right hand rule to choose the next face for traversal. GPSR is not efficient if it can not find the correct face quickly; and in the worst case, it traverses all the bad faces and finds the correct one last.

To improve the performance of face routing, Kuhn et al. propose a Greedy Other Adaptive Face Routing (GOAFR) protocol [5]. GOAFR uses an adaptive face routing (AFR) mode if the greedy mode encounters a void area. The basic idea of AFR is to adjust the boundary of a traverse ellipse area around the face and choose an optimal value to reach the destination. The boundary is decided by the Boundary Face Routing (BFR) that uses the same rule of face routing except that the exploration around a face will walk back when it reaches the boundary. If a packet can not reach the destination via BFR, it will be routed back to the source node [5]. The boundary is then doubled, and the process is repeated again until the destination is reached.

Both the GPSR and GOAFR protocols planarize network topology to support face routing and the planarization (GG or RNG [4]) algorithms assume that the connectivity between nodes can be described by unit graphs, i.e. a node is always connected to all neighbors within its fixed transmission range while not connected to nodes outside this range. In an experimental deployment of GPSR protocol in wireless sensor networks by Kim et al. [6, 7], they have observed that there exists permanent routing failures because the unit-graph assumption cannot be satisfied in practical scenarios. To solve this problem, Kim et al. [6] propose a distributed Cross-Link Detection Protocol (CLDP) to planarize the network. However, CLDP is complex and costly because it induces new routing overhead caused by "probe" packets for planarization.

To reduce the overhead of CLDP, Kim et al. [13] propose a new approach, Lazy Cross-Link Removal (LCR) for geographic routing. LCR removes non-planarities lazily only when a loop is detected in the face routing. LCR can results in an order of magnitude or more lower overhead than any previously proposed approach for two reasons. First, greedy routing does not reach local maxima in most cases so that many non-planarities are masked because they never cause routing failures. Second, one needs only eliminate all cross-links that induce looping face walks to ensure planarity, and very few cross-links actually induce such packet loops. LCR still uses face routing and planarization algorithms to recover local maxima. LCR targets at applications in static sensor networks.

In [9], Leong et al. present a new geometric routing protocol without using network planarization, i.e. Greedy Distributed Spanning Tree Routing (GDSTR). The GDSTR protocol generates spanning tree(s) and every node maintains a convex hull based on its children in the spanning tree and their convex hulls. When a node can not forward a packet using the greedy algorithm, it switches to the recovery mode and checks if the destination is contained in its convex hull and decides whether to forward the packet to a proper child or just send it to its parent that has a bigger convex hull. GDSTR can achieve better hop stretch and path stretch than CLDP with less overhead. However, GDSTR is proposed for static sensor networks and not stateless. Therefore it is not suitable for MANET because the convex hull maintenance is costly in dynamic scenarios.

Similar to GDSTR, our proposed protocol, LTGR, does not use planarization either. However, LTGR differs from GDSTR in the fact that it is stateless and does not need any global information or extra message exchange to recover routing from void areas. Like GOAFR, LTGR keeps the adaptability of routing exploration in the recovery process, i.e. LTGR selects a sub-tree adaptively for packet forwarding based on the position information of the leaf nodes in each sub-tree when it routes packets. Because the selection utilizes position information, instead of a constant boundary value used in GOAFR, LTGR can make better routing decisions than GOAFR.

III. Local Tree Based Geometric Routing (LTGR)

A. *Overview of LTGR*

Like existing geometric routing protocols, LTGR takes advantage of the greedy algorithm to route packets whenever possible. A packet can be either routed in the greedy mode if the greedy algorithm works or in the recovery mode if the packet routing reaches a void area. A flag in the packet header marks the routing mode of a packet. LTGR uses a local tree based routing algorithm to route packets in the recovery mode to bypass void areas. Whenever a node receives a recovery-mode packet, it checks whether it is closer to the destination than the originator of the recovery process, and if positive, it switches the packet back to the greedy mode and uses the greedy algorithm to forward the packet. This process is repeated until the destination is reached, or all the possible paths are tried once and still no route is found to reach the destination, e.g. when the network is partitioned.

If a packet is routed in the greedy mode, it would not encounter any routing loop. If the packet is routed in the recovery mode, the local tree information used by the LTGR to bypass the void area will be embedded in the packet header, thus any node can get the history information of the tree to avoid forming routing loops. The local tree could expand to a spanning tree covering all nodes in the network. Therefore, LTGR can guarantee the packet delivery if there is a path between the source and the destination.

B. *Search Algorithms*

LTGR uses local tree based search algorithms to find paths bypassing void areas so that packets in the recovery mode could be routed.

In this research, we first study the uniform cost search in the situation that the source node knows nothing about the whole topology and the destination's position. For example, the breadth first search, in which all nodes at level d are expanded before any nodes at level $d+1$, finds the shallowest goal state, i.e. the shortest path. If we define the function DEPTH(n) as the depth of the node n, then the node with the lowest DEPTH(n) value is always expanded first. If the route cost is a function of the depth of the solution, e.g. the hop count of the route, the breadth first search can achieve the best solution, i.e. the optimal path.

Although the uniform cost search can find the optimal route provided that there is no negative cost, it would traverse most of the possible routes, which could induce a large amount of overhead. Assume B stands for the average branching factor, and D is the depth of the solution, the complexity of the uniform cost search is $O(B^D)$. This overhead is prohibitive in large scale MANET.

To address the overhead issue, we consider the greedy search algorithm. A node using the greedy search algorithm always finds a neighbor node as the next hop that is the closest to the destination among all neighbor nodes. Thus the greedy algorithm can select the next hop exclusively therefore eliminates the overhead of traversing other possible route in the uniform cost search algorithm. Moreover, it is faster than the uniform cost search algorithm in average.

However, the greedy search algorithm is efficient but not complete, which can not guarantee finding an existing path to the destination because no history information is recorded. The uniform cost search is complete because it records history information; and it can find the optimal path but is not as efficient as the greedy search. Thus it is desirable to combine them together for path searching.

In this research, we augment the greedy search algorithm with the history information and propose the local tree based search algorithm.

A tree consists of a root, branch nodes and leaf nodes. The first node N_0 that routes a packet reaching a void area marks the packet to be in the recovery mode and initializes the recovery process and the local tree: it sets itself as the root of the local tree and its neighbors as the leaf nodes. After the tree is constructed, the tree information is stored in the packet header and forwarded along with the payload to the next-hop node N_1 that is a leaf node of the tree closest to the destination according to the greedy search algorithm. When a leaf node, say N_1, receives the packet, it first retrieves the local tree information from the packet header and checks if it is closer than the root to the destination. If so, the routing mode of the packet will be switched back to the greedy mode and the tree information will be removed from the packet header. Otherwise, the node N_1 expands the local tree by adding all its neighbors as its children; and thus it is changed from a leaf node to a branch node. The information of the expanded local tree is stored in the packet header and the packet is forwarded to a leaf node of the updated tree, say N_2, which is the closest one to the destination among all leaf nodes based on the greedy search algorithm. Note that N_2 may not be the neighbor of N_1 but N_1 can find a path to N_2 based on the local tree information.

The local tree based search algorithm is complete since it can guarantee the packet delivery if there is a path to the destination.

C. LTGR Protocol

Based on the LTGR protocol, a node in MANET routes a packet by the greedy algorithm if the packet does not encounter a void area. Otherwise the packet will be routed in the recovery mode of the LTGR protocol.

Because LTGR uses a tree structure, there is no loop in the path. And in the worst case, the local tree will expand to a spanning tree that can reach every node in the network. The challenge of embedding local tree information in the packet header is that the header overhead may be very large in dense networks. To address this challenge, we propose two techniques used in LTGR.

First, each node divides the network space into four quadrants when it receives a recovery mode packet. The division is based on the axis x, which is the line connecting itself with the destination, and axis y, which is the line perpendicular to axis x and passing this node. After that, the node adds only three neighbors to the local tree, which are distributed in the three quadrants (except the quadrant that contains the previous hop node) and are closest to the destination among the neighbors in each quadrant. This means that each branch node in the local tree only has maximum three children no matter how dense the network is and only the root has maximum four children from all four quadrants. The above-mentioned process may need to be repeated to guarantee the packet delivery when the network is not partitioned if a packet could not be routed to the destination in the first round.

Second, we propose a new compression technique to compress the local tree information stored in the packet header. For the tree structure, we only use 2 bits to represent the structure for each branch node because it only has 3 children. For the node information, because only the to-destination-distance values of the root and the leaf nodes are useful in the recovery mode, we do not need to include theses values of branch nodes in the packet header. Comparing to face routing algorithm that keeps the position information of all the nodes along a face, LTGR has much less overhead. Suppose that there are n nodes in total and m leaf nodes in a local tree, and the node id and the to-destination-distance value are represented by k bits and t bits respectively, the total bits that are needed to store the tree information is:

$$T(n, m, k, t) = 2\,(n-m) + k \times n + t\,(m+1) \qquad (1)$$

An example that uses LTGR to recover the void area is illustrated in Fig. 1. A is the sender and D is the destination. First, A sends a packet to S using the greedy algorithm because S is closer than A to D. When S receives the packet, it initializes the recovery process because it has no neighbor closer to D than itself. It constructs a local tree by setting itself (S) as the root and adding three neighbors F, B, and E as leaf nodes, which are closest to D in the quadrants I, II, IV respectively. Then the packet is forwarded to B because B is the closest to D among all the leaf nodes. When B receives the packet, it adds C to the local tree. However, B finds that the leaf node with the shortest distance to D is F, instead of C or E. And it forwards the packet to F through the path $B \rightarrow S \rightarrow F$. After that, when F receives this packet, it adds its neighbors H and G to the local tree, and forwards the packet to H that is the closest to D among all the leaf nodes. At last, H receives the packet and finds it is closer than the root S to the D. It changes the packet to the greedy mode and forwards the packet using the greedy algorithm.

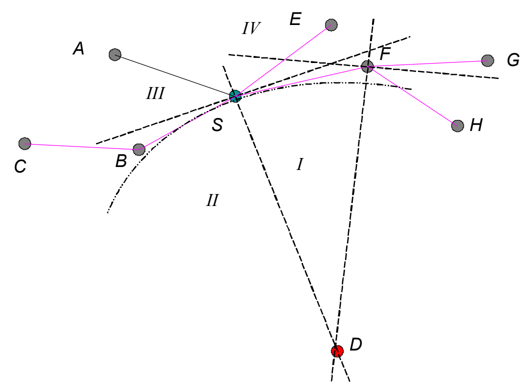

Figure 1. A local tree based routing example.

IV. Simulation and Result Analysis

To validate our protocol, we have implemented LTGR in ns-2(.28) and simulate the protocol using various mobile ad-hoc network topologies. We have also tested our protocol with different traffic patterns. LTGR is compared with GPSR whose source code is downloaded from its authors' website [11]. We do not compare LTGR with CLDP and GDSTR since they are designed for static sensor networks and not suitable for highly dynamic MANET. Although there are numerous metrics to evaluate a routing protocol design for MANET, we focus mainly on the packet delivery ratio, routing overhead and average hop stretch achieved by the routing protocols.

A. Simulation Setup

We use the same parameters that are listed in Table I for both LTGR and GPSR simulations.

TABLE I. Simulation Parameters

Param.	***Beacon Interval***	***Trans. range***	***Position variable size***	***Network size***	***Simu. Time***
Value	1s	250m	12 bits	1000×1000 (m^2)	900s

- Movement model: Nodes move according to the "random waypoint" model [1]. We observe that the probability that a packet would be routed in the recovery mode is inversely proportional to the pause time (Fig. 2). Because both protocols use the greedy algorithm whenever possible, we set the pause time as 0 to compare their performance in the recovery process. We use CMU scene generator to generate 80 different pattern files based on 8 different numbers of nodes: 20, 30, 40, 50, 60, 70, 80 and 90 (10 files for each number respectively). The moving speed of nodes is distributed uniformly between 1 m/s and 20 m/s.

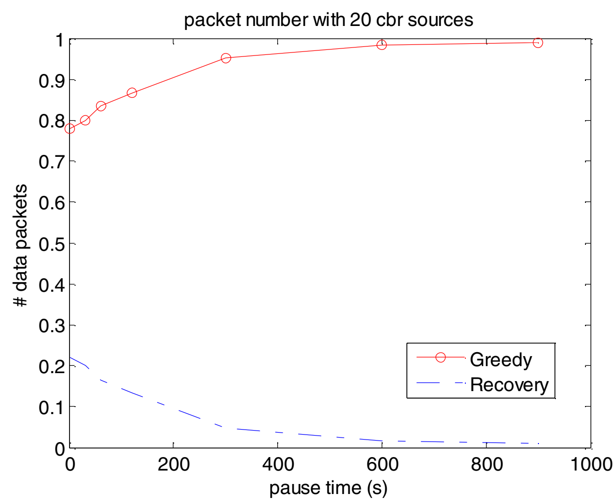

Figure 2. Percentage of packets in recovery mode as a function of pause time

- Traffic pattern: we choose UDP as our transport layer protocol. Randomly selected 14 nodes generate 20 traffic flows. The transmission rate of every flow is 1Kbps: one 512 bytes packet is generated every 4 seconds. The starting time instances of the traffic flows are uniformly distributed between 0 and 180 seconds.

Both LTGR and GPSR are simulated based on all the above-mentioned 80 various scenarios with the described traffic pattern and movement models; and the average values are calculated and studied.

B. Packet Delivery Ratio

Packet delivery ratio is the number of packets received by the destinations divided by the number of packets originated from the sources in the application layer. It describes the loss rate of networks and characterizes both the completeness and the correctness of a routing protocol [1].

Both protocols can achieve good delivery ratio and there is only slight difference between them as shown in Fig. 3. This is because both protocols can guarantee the packet delivery if there is a path between the source and the destination. LTGR performs slightly better than GPSR. In fact, some packets are dropped by LTGR because of ARP errors that are caused by neighbors' mobility, while an extra number of packets are dropped by GPSR due to TTL (set to be 128 in both protocols) errors in that the face routing may cause infinite routing loops in the dynamic scenarios. Because both LTGR and GPSR use the greedy algorithm when it works, we conclude that the local-tree based routing can achieve a higher success ratio than face routing when packets bypass void areas in the recovery mode.

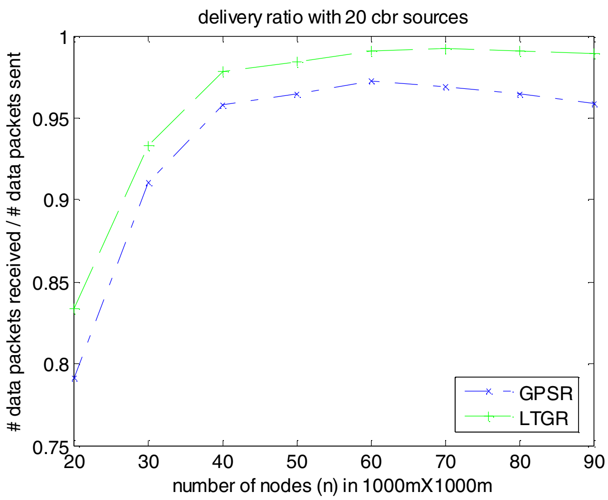

Figure 3. Packet delivery ratio as a function of the number of nodes.

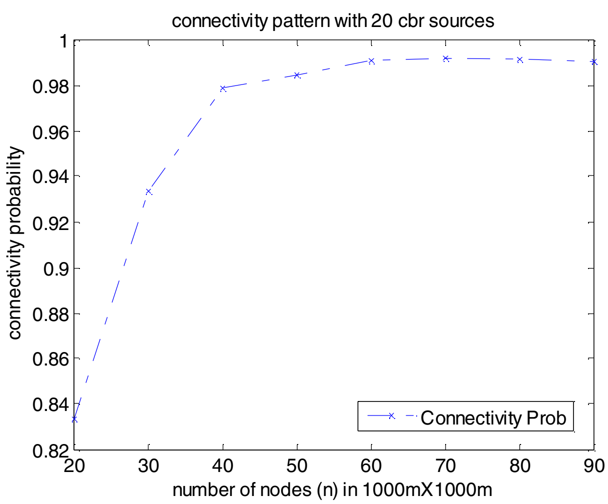

Figure 4. Source-destination connectivity probability as a function of the number of nodes.

The reason that the delivery ratio increases along with the increase of the node density is because the connectivity probabilities of the source nodes and the destination nodes increase when the node density increases, as shown in Fig. 4.

C. *Average Hop Stretch*

The average route hop stretch stands for the route optimization degree of a routing protocol. By establishing a shorter route, a routing protocol can take advantage of the network resources more efficiently. In this research, we define hop stretch as the ratio of the real hop count that a packet passes from the source to the destination to the hop count of the optimal path. Because both protocols use the greedy algorithm whenever possible and most packets can reach the destination by only using the greedy algorithm as the shown in Fig. 2, it is undesirable to compare the average hop stretch of all transferred packets. In our simulation, we mark every packet that has ever been delivered in the recovery mode (via the local tree algorithm or the face routing algorithm), and only compare the average hop stretch of these packets.

The average hop stretch of LTGR is much better than that of GPSR as shown in Fig. 5. GPSR uses the face routing algorithm to bypass void areas. However, the face routing is not efficient because of the following reasons. First, it uses right hand rule to choose a face blindly. Second, it has to complete the face if it chooses a wrong face before changing to the next face. Third, in the planarization process, some shorter paths would be deleted, e.g. the diagonals of a full connected rectangle would be deleted to planarize the graph. LTGR is efficient because it utilizes the position information to decide the next hop, i.e. the greedy search uses the to-destination-distance values of each leaf node. Because LTGR does not use planarization, it also keeps the shortest path to the destination.

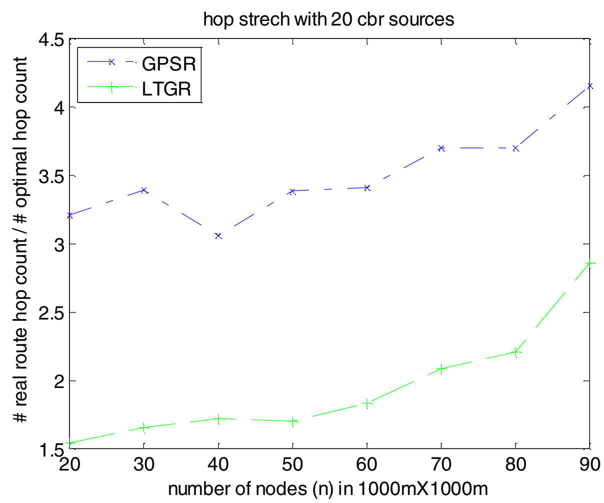

Figure 5. Average hop stretch as a function of the number of nodes.

The average hop stretches of both protocols roughly increase as the node density increases. This is because the number of possible paths to the destination increases as the node density increases. Thus the probability of selecting a right but not the optimal path increases too.

D. *Protocol Overhead*

Routing overhead is another important metric in routing protocol comparisons because it implies the efficiency of a protocol in terms of bandwidth consumption and battery power usage. Large routing overhead induces the congestion in a low-bandwidth environment and harms the scalability of the network. Here we define the routing overhead as the number of bytes sent by all nodes divided by the number of payload data bytes received by the destinations. The overhead does not include IEEE 802.11 RTS/CTS packets or ARP packets. However, it does include the overhead of the IP header because we modify the IP header in LTGR to store the position information of the destination as what is done in the GPSR. Thus the IP header in LTGR implementation could be larger than those in other ad-hoc routing algorithms, such as AODV. The routing overhead performances of GPSR and LTGR are shown in Fig. 6.

The overheads of both protocols generally decrease as the node density increases. This is because the connectivity probability of the network increases as the node density increases as shown in Fig. 4. If the network is partitioned, both algorithms would try all the possible paths before dropping the packets, which generates more overheads.

There are two features of LTGR that contribute to its better overhead performance than that of GPSR. One is that the delivery ratio achieved by LTGR is higher than that of GPSR.

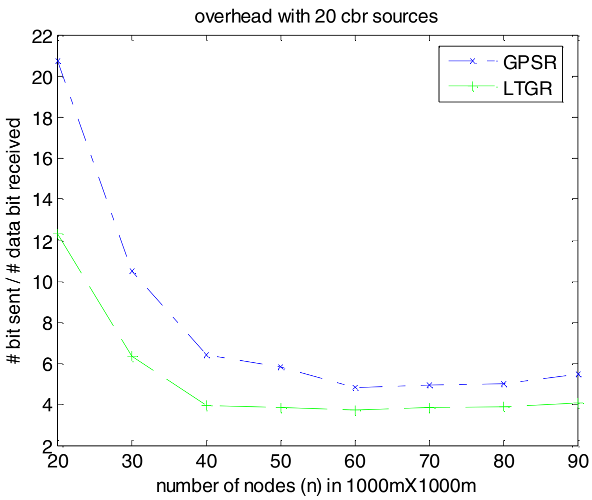

Figure 6. Protocol overhead as a function of the number of nodes.

Because GPSR drops extra packets due to TTL errors, these packets generate extra overhead. The other is that LTGR is more efficient than GPSR in terms of hop stretch, which means that the local tree based routing algorithm in LTGR can bypass the void area quicker than the face routing algorithm in GPSR. A longer search process for a correct path requires the packets to be delivered to more invalid path candidates that cause more overhead.

V. CONCLUSION

In the existing literature, a number of geometric routing protocols that are based on the face routing have been proposed for MANET. However, the planarization used in the face routing has implementation problems observed by testbed experiments. In this paper, we propose a stateless geometric routing protocol LTGR to overcome the shortcomings of the face-routing based protocols. LTGR uses a local-tree based search algorithm to choose the next-hop node to bypass void areas when the greedy algorithm does not work. The tree based search is complete and LTGR can route packets bypassing void areas and avoid forming routing loops. Comparing to GPSR, LTGR is more efficient in terms of routing overhead and hop stretch shown by extensive simulation results. For example, LTGR can reduce the routing overhead by 25 ~ 40% and the hop stretch by 30 ~ 50% comparing to GPSR in our simulation scenarios.

ACKNOWLEDGEMENTS

This research has been supported by the National Science Foundation (NSF Award #0438300) and a grant from the Commonwealth of Pennsylvania, Department of Community and Economic Development, through the Pennsylvania Infrastructure Technology Alliance (PITA). Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the sponsors of this research.

REFERENCES

[1] J. Broch, D. A. Maltz, D. B. Johnson, Y. C. Hu, and J. Jetcheva. “A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols.” *Proceedings of ACM/IEEE Mobile Computing and Network*, pp. 85-97, Dallas, TX, October 1998.

[2] E. M. Royer and C.-K. Toh. “A Review of Current Routing Protocols for Ad-Hoc Mobile Wireless Networks.” *IEEE Personal Communications Magazine*, Vol. 6, No. 2, pp. 46-55, April 1999.

[3] V. Ramasubramanian, Z. J. Haas, and E. G. Sirer. “SHARP: A Hybrid Adaptive Routing Protocol for Mobile Ad Hoc Networks.” *Proceedings of the ACM Symposium on Mobile Ad Hoc Networking and Computing* (*MobiHoc*'03), pp. 303-314, Annapolis, Maryland, June 2003.

[4] B. Karp and H.T. Kung. “GPSR: Greedy Perimeter Stateless Routing for Wireless Networks.” *Proceedings of the Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking* (*MobiCom* 2000), pp. 243-254, Boston, MA, August 2000.

[5] F. Kuhn, R. Wattenhofer, and A. Zollinger. “Worst-Case Optimal and Average-case Efficient Geometric Ad-hoc Routing.” *Proceedings of the Fourth ACM International Symposium on Mobile and Ad hoc Networking & Computing* (*MobiHoc*'03), pp. 267-278, Annapolis, Maryland, June 2003.

[6] Y.J. Kim, R. Govindan, B. Karp, and S. Shenker. “Geographical Routing Made Practical,” *Proceedings of the 2nd Annual Symposium on Networked Systems Design and Implementation*, pp. 217-230, Boston, MA, May 2-4, 2005.

[7] Y. Kim, R. Govindan, B. Karp and S. Shenker. “On the Pitfalls of Geographic Face Routing.” *Proceedings of the 2005 Joint Workshop on Foundations of Mobile Computing* (*DIALM-POMC'05*), pp. 34-43, Cologne, Germany, September 2005.

[8] Y. Zhang and L. Cheng. “PLACE: Protocol for Location And Coordinates Estimation -- A Wireless Sensor Network Approach.” *Computer Networks* (Elsevier), Vol. 46, No. 5, pp. 670-693, December 2004.

[9] B. Leong, B. Liskov, and R. Morris. “Geographic Routing without Planarization.” *Proceedings of the 3rd Symposium on Network Systems Design and Implementation (NSDI 2006)*, San Jose, CA, May 2006.

[10] E. Kranakis, H. Singh, and J. Urrutia. “Compass Routing on Geometric Networks.” *Proc. 11th Canadian Conference on Computational Geometry,* pp. 51-54, Vancouver, Canada, August 1999.

[11] http://www.cs.cmu.edu/~bkarp/gpsr/gpsr.html. Accessed on 02/26/2007.

[12] B. Karp. “Challenges in Geographic Routing: Sparse Networks, Obstacles,and Traffic Provisioning.” *DIMACS Workshop on Pervasive Networking*, http://www.cs.cmu.edu/~bkarp/geochal-dimacs2001.pdf, Piscataway, NJ, May 2001. Accessed on 02/26/2007.

[13] Y. K. Govindan, B. Karp, and S. Shenker. “Lazy cross-link removal for geographic routing.” *Proceedings of the 4th international Conference on Embedded Networked Sensor Systems* (SenSys'06), November 2006.

[14] I. Stojmenovic and X. Lin. “Loop-free Hybrid Single-path/Flooding Routing Algorithms with Guaranteed Delivery for Wireless Networks”. *IEEE Transactions on Parallel and Distributed Systems*, Vol. 12, No. 10, pp. 1023–1032, 2001.