HTL: A Locality Bounded Flat Hash Location Service*

Ruonan Rao, Shuying Liang, and Jinyuan You

Department of Computer Science and Engineering
Shanghai Jiaotong University Shanghai, China 200030
rao-ruonan@cs.sjtu.edu.cn,liangsy@sjtu.edu.cn,you-jy@cs.sjtu.edu.cn

Abstract. Many location services have been proposed, but some challenges remain. In this paper, we present a new location service, named HTL (Hash Table Localized) to solve the locality problem, that in a location service, the location information can be stored potentially far away from both the source and destination nodes, even when the source and destination nodes are close. As a result, it causes high overhead in update and query. HTL uses a double index hash function to map a node to a location in the network area called the virtual coordination of that node. Then, a novel method is employed to divide the physical space into lattices. The publish and query algorithms are designed based on this division. In HTL, when the distance between the source and destination nodes is l, the cost of query is $O(l^2)$. We define this property as n^2-locality bounded. HTL is the location service that achieves this property with the least storage and network overhead. Both analysis and experiment results are presented in this paper concerned with the cost, the locality bounded property and the scalability.

1 Introduction

A mobile ad hoc network (MANET) is a network formed by a collection of mobile wireless nodes without any perviously deployed infrastructure. Due to the lack of infrastructure support, each node in MANET should act not only as an end system but also as a router at the same time.

A fundamental challenge in MANETs research is the design and implementation of scalable and robust routing protocols. The current routing protocols can be roughly divided into two categories: Topology-based routing and Geographic-based routing. The former is based on the knowledge of the whole network's topological information while the latter on the knowledge of each node's position. Intuitionally, compared with topological-based routing, geographic-based routing incurs less communication overhead. However, geographic-based routing also faces two challenges:

1. How to deliver a packet to destination when the position of destination node is known? This is called as the forwarding strategy.

* This paper is supported by the Defense Pre-Research Foundation of China under Grant No.513150302

J. Cao et al. (Eds.): NPC 2008, LNCS 5245, pp. 138–149, 2008.

2. How to let the source node know the position of the destination node? This is called as the location service.

The first one is almost solved by several proposed algorithms, especially GPSR[1] [2]. Whereas, some problems remain in the second one, though many location service algorithms have been proposed. One of them is called locality problem, i.e. the corresponding location information may be stored far away from both the source and destination nodes, but the source and destination may be close. As a result, update and query operations cause high overhead. This problem is more serious in a location service without any hierarchy architecture, which we called, the flat location service. Compared with hierarchy approach, a flat location service avoids the complexity of maintaining the hierarchy structure, but introducing the locality problem. In this paper, we try to solve this problem.

HTL is a flat hash based location service, which means that HTL uses hash functions to determine where to store the nodes location information, and no hierarchy in HTL. A hash based approach uses hash functions to map a node to a location or a region in the network area, then stores the corresponding location information near or in such location or region. HTL is designed to cooperate with GPSR to support geographic based routing. With a novel method to divide the geographic area into lattices, HTL guarantees that when the distance between the source and destination nodes is l, the query takes at most $O(l^2)$ to finish. HTL is the best one we have known that takes the least cost to achieve such a property.Some simulation experiments are used to verify HTL.

This paper has 7 sections. It starts with an overview of related works in section 2. In section 3, we state the locality problem of location services in MANET. In section 4 and section 5, details of HTL and some mathematical analysis are given. The experiments results are presented in section 6. Section 7 is a short conclusion with discussion on future works.

2 Related Works

Many location service algorithms have been proposed. Surveys on some of the algorithms can be found in [3], [4] [5] [6] and [7], but not all of them are hash based. The following are some typical algorithms proposed related to location service.

GLS [7] is a location service with a hierarchy of square regions. A node belongs to only one square in each order in the hierarchy. It stores its location information on 3 nodes in each square containing it. If two nodes are close enough, they will be in the same square area with a low order and need not travel a long distance to exchange the location information. However, the cost of maintaining such a hierarchy structure is expensive. The work in [8] shows some similar results. In addition, GLS has not been proved that it has an upper bound of query cost. LLS[9] is the first location service that takes locality into account. It uses a recursive approach that promulgates the location information to the nodes 2^i away in ith step, with a similar query method. It has been proved that it is d^2 locality bounded. However, the cost of publishing location information and storing the

location information is relatively high. GHLS protocol proposed in [10] is a simple flat hash based location service. In this paper, the authors have mentioned the locality issues, and tried to solve those problems by a method called α-scaled region– a hash function that maps a node to a location only in a region called **scaled location server region** which is similar to the whole area and located in the center. Intuitively, this approach can reduce the cost of query when the source and destination nodes are all near the center, but its effect is limited in other situations. In addition, GHLS does not possess the locality bounded property, either. GHT[11] is designed for data-centric storage in sensornets. One can consider location information as a specifical data in sensornets, and augments of GHT can also be used as a location service. Although GHLS and our work share some characteristics with GHT, GHT can not be effectively used as a location service in MANET. The design objectives in GHT are fundamentally different from location services. An analysis on GHT in MANET can be found in [12].

3 Locality Problem in Flat Hash Based Location Services

A location service is a service that provides the corresponding location information according to a node's unique identifer. Its main functionality is to support geographic based routing, when it also can be used to support location related applications.

In a typical flat hash based location service, a hash function is used to map each nodes unique identifier to a home region. However, the cost of underlying routing can be surprisingly high if the source and destination nodes are close, when the home region is far away. To formally evaluate this problem, following definition is given.

Definition 1 (Locality Bounded). *A location service is called* ***locality bounded****, if the distance between the source node S and the destination node D is l, and the cost of query the location of D by S is at most $f(l)$. We call such a location service is f-Locality Bounded.*

4 HTL: A Hash Table Localized

In this section, we present the details of HTL. HTL is built on the top of GPSR, and it cooperates with GPSR to provide geographic based routing. We first study some properties of GPSR, then some concepts of HTL are introduced, followed with a novel method by which HTL uses to divide the physical space being presented. Finally,the publish and query algorithms are illustrated.

4.1 Map to Physical Space

As stated before, a flat hash based location service will map a node's identifer to a region named home region using a hash function. HTL uses the same approach with a little modification. The hash function used in HTL is a double index hash

function $H(N.id) = (h1(N.id), h2(N.id))$ that maps node N's identifier $N.id$ to a coordination $(h1(N.id), h2(N.id))$ inside the area of the network. We call the associated coordination $(h1(N.id), h2(N.id))$ of node N N's virtual coordination. The definition is given in definition 2.

Definition 2 (Virtual coordination). *A Node N with unique identifer $N.id$, its virtual coordination is $H(N.id) = (h1(N.id), h2(N.id))$, where $h1$, $h2$ is hash functions.*

The major difference between typical flat hash based location services and HTL is that HTL pushes the concept of home region to the extreme: the home region becomes a single point in this area.

4.2 Divide the Physical Space

With node N's identifier $N.id$, one can find the virtual coordination $H(N.id)$. We define a parameter d named lattice length. After finding the virtual coordination, HTL divides the physical space into lattices using the virtual coordination as the original point according to the following rules:

- Make circles C_i with radius $r_i = d * i, i = 0, 1, \ldots$[1] with the same center $H(N.id)$. We call these circles **lattice circle**s.
- For a circle C_i, $(i = 1, 2, \ldots)$, let $j = \lceil \log_2 i \rceil$, then divide C_i into 2^{j+2} with equal angle, starting from the x axis. We denote the lines used to divide the circle as $l_m^{2^{j+2}}, m = 0, 1, \ldots, 2^j - 1$, and call them **lattice line**s. These lines intersect the circles $C_k, k = 2^j, 2^j + 1, \ldots, i$ with points P_m^k, which are named **lattice point**s.
- For each 4 points, P_k^m, P_k^{m+1}[2], P_{k+1}^m, P_{k+1}^{m+1}, $(m = 0, 1, \ldots, 2^j - 1, k = 2^j, 2^j + 1, \ldots, i - 1)$, form a **lattice corner**, dedicated by LC_k^m.
- The area enclosed by lattice corner LC_k^m and the corresponding line and circle is called a **lattice**, denoted as L_k^m.

4.3 Location Servers Selection and Location Information Update

Unlike other typical flat hash based location service, HTL stores the location information not only in the home region, but also stores on the way to the home region. For a node N, assuming its current location is $N.loc$, and its virtual coordination is $H(N.id)$. Obviously, a short path that travels along the lattice lines exits between $N.loc$ and $H(N.id)$. The path can be found based on following steps:

1. First, determine the lattice $L(N.loc)$ where N is based on N's current location $N.loc$:

[1] When $i = 0$, the circle becames the point $H(N.id)$.

[2] Here, $m+1$ means $m+1 mod 2^{j+2}$. In this paper, we use m_n to denote such situations when it is not ambiguous.

(a) Calculate the distance $dist$ between $N.loc$ and $H(N.id)$. Let $k = \lfloor dist/d \rfloor$ and $j = \lceil \log_2 i \rceil$.

(b) Calculate the angle α from vector $\overrightarrow{(1,0)}$ to vector $\overrightarrow{(N.loc, H(N.id))}$ in anticlockwise direction. Let $m = \lfloor \frac{\alpha}{\frac{2\pi}{2^{j+2}}} \rfloor = \lfloor 2^{j+1} \frac{\alpha}{\pi} \rfloor$.

(c) Then node N is located lattice $L_(k)^m$. Two corresponding lattice lines are $l_m^{2^{j+2}}$, $l_{m+1}^{2^{j+2}}$.

2. One of the lattice lines $l_m^{2^{j+2}}$, $l_{m+1}^{2^{j+2}}$ is closer to N than another [3]. The location information will be sent along this line.
3. When the path comes to the circle $C_{2^{j-1}}$, the lattice line chosen will terminate. However, it may not arrive the virtual coordination $H(N.id)$. Then a new lattice line should be chosen. If the current lattice line is $l_{m'}^{2^{j'}}$, the next lattice to travel is chosen based on the following rule:
   - If m' is an even number, the next lattice line is $l_{m'/2}^{2^{j'-1}}$
   - If m' is an odd number, the next lattice line is $l_{(m'-1)/2}^{2^{j'-1}}$

   This rule can be not only applied when the first lattice line is terminated, but also suitable for the next lattice line and next's next lattice line.

The shortest path to the virtual coordination can be found. But where the location information will be stored? In each lattice line along the path found, there are several lattice points, called **server coordination**s. The location servers are the nodes nearest to these server coordinations. The location information of N will be stored on these nodes.

Then another problem arises—how to deliver the packet to the nodes that are nearest to the server coordinations? We use GPSR routing to solve this problem. Some properties of GPSR can help to decide when and who will consume the packet for location disseminating. Thus a location server can be easily found during the routing process. The next question to answer is when should a node publish its location information into HTL, and how?

When Node N moves from location x to location y, it performs the publish as following:

- If $LN(x) = LN(y)$, new location information is published to the same servers.
- Else, node N first issues a packet to infor the servers that store the old location information to erase the information. Then it publishes the new location information.

Another parameter d_u is defined to indicate the distance a node can move before it do a update. Generally speaking, when a node move from one lattice to another, it must perform an update. As a result, d_u should not be larger than $d/2$, where d is the lattice parameter defined above.

[3] If the distances from N to the two lines are equal, $l_m^{2^{j+2}}$ is chosen.

4.4 Perform Query

When node S wants to know where D is, it performs a query operation. The query operation performs in tow modes: extension mode and non-extension mode. In extension mode, the query will goes to the servers located on the adjacent lattice lines, when in non-extension mode, the query is limited to current lattice lines. The query process try to cover the area that may intersect with the path where D puts its location information as soon as possible. The details of the query process is described bellow.

1. In step 1, S calculates which lattice of D it is in using the method given in previous section. We denote this lattice as L_k^m. Node S first query the location servers whose location server coordination are P_k^m and P_k^{m+1} in **extension mode**(P_k^m left-extension and P_k^{m+1} right-extension).
2. In step i ($i = 1, 2, 3, \ldots, k-1$), location servers $P_{k-i+1}^{m'}$ [4] will get the query. It process the query based on the following situations:
   - It knows where D is, then it answer this query, and informs other location server in the same iterative level to terminate this query.
   - It dose not know the answer.
     - The query is in non-extension mode.
       - $k - i + 1 == 2^j$, no future action is needed.
       - $k-i+1! = 2^j$, forward the query to $P_{k-i}^{m'}$ in non-extension mode.
     - The query is in extension mode. We only discuss the left-extension case. The right-extension case is similar.
       - $k - i + 1 == 2^j$ and m' is an odd number. Deliver a query to $P_{k-i}^{\frac{m'-1}{2}}$ in left-extension mode.
       - $k - i + 1 == 2^j$ and m' is an even number. Deliver a query to $P_{k-i}^{\frac{m'}{2}-1}$ in left-extension mode and a query to $P_{k-i}^{\frac{m'}{2}}$ in non-extension mode.
       - $k - i + 1! = 2^j$, Deliver a query to $P_{k-i}^{m'-1}$ in left-extension mode and a query to $P_{k-i}^{m'}$ in non-extension mode.
3. In step k, the query arrives $H(D.id)$. If the location server knows where D is, then it reply the answer, otherwise it will tell S the location of D is not known.

Figure 1 gives an example of query in HTL. Node S wants to know where D is. It calculates the lattice of D where it is in. In the example, S is in L_6^{19}, it deliver queries to P_6^{19} in left-extension mode and to P_6^{20} in right-extension mode. We trace the thick line in figure 1 to explain the query process in detail. In the first iterative step, P_6^{19}, 6 is not the power of 2. P_5^{18} (left extension mode), P_5^{19} (non-extension mode) will get the query. Then, P_4^{17} (left extension mode), P_4^{18} (non-extension mode), P_4^{19} (non-extension mode). Here, $4 = 2^2$, and 17 is an odd number. Then, P_4^{17} deliver query to P_3^8. The process will go on until a server replies or the query arrives $H(D.id)$.

[4] We use the location server coordination to denote location server, when there is no ambiguousness.

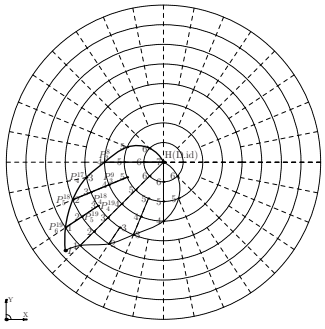


Fig. 1. An Example of Query in HTL

4.5 Location Servers Maintain

Due to the node mobility, location information stored on one location server may need to be migrated to other nodes that become closer to the server coordination. As in GHLS[10], every node acts as a location server to check whether there is a neighbor node closer to the server coordination when it receives a beacon packet from another node. GPSR uses beacon to construct the location information about its neighbor nodes. HTL uses this facility of GPSR to maintain the location servers.

5 Analysis on HTL

Due to space limitation, we omit some properties of HTL here, but remain the most important one, the Locality Bounded Property. A further analysis for this property is also shown.

Query Cost and Locality Bounded Property: As stated before, the most remarkable feature of HTL is that HTL is l^2 locality bounded.

Theorem 1. *HTL is l^2 locality bounded.*

The proof is not presented here due to the limitation of space.

A brief Comparison: Table 1 gives a brief comparison between HTL and other location service related in section 2. For detailed analysis, please refer [12],[10] and [13]. From the table, we can conclude that only LLS and HTL bear the property of locality bounded. Yet compared with LLS, HTL is with less network and storage overhead.

6 Experiments

In this section, we present the experiments results on HTL. The experiments are concerned with the performance of HTL in static network, and in mobile network and with the issue of scalability.

Table 1. A brief Comparison

	GLS	GHLS	GHT	LLS	HTL
Cost of publish	$O(h^2)$	$O(\sqrt{N})$	$O(\sqrt{N})$	$O(u^2)$	$O(\sqrt{N})$
Cost of Storage	$O(h)$	$O(1)$	$O(Dens)$	$O(u^2)$	$O(\sqrt{N})$
Server Maintaining	$O(2^h)$	$O(1)$	$O(N)$	N/A	$O(1)$
Locality Bounded	N/A	N/A	N/A	$O(l^2)$	$O(l^2)$

In GLS the height of grid is set to h, and in LLS the unit length is u. N is the number of nodes in the network.

6.1 Experiment Method and Metrics

We have implemented HTL in ns2[14]. In all experiments, we use a 802.11 radio model with a nominal bit-rate of 11Mbps and a transmission range of 250m. The discrete metrics are listed in table 2. The experiments on static and mobile networks run 300s, and the experiments on scalability run 150s.

Table 2. Metrics used for evaluating HTL

Metric	Description
P_n	The number of packets used to publish location information. We call such packets PUT packets.
P_r	The average hops a PUT packet travels.
R_n	The number of packets used to erase location information. We call such packets REMOVE packets.
R_r	The average hops a REMOVE packet travels.
U_n	The number of packets used to migrate location information to another node. We call such packets UPDATE packets.
U_r	The average hops a UPDATE packet travels.
LN_{max}	The maximum number of location information entries a node stored.
LN_{min}	The maximum number of location information entries a node stored.
LN_{ave}	The average number of location information entries a node stored.
RQS	The ratio of query success.

All experiments are performed as following:

- In the first a few second (30s), all nodes publish their location information into HTL.
- Then, each node chooses a destination randomly and independently, and queries its location.

Concerned with static networks, the topology scenarios are generated using the following model:

1. Uniform and Random Network. If there are N nodes in the network, then we divide the whole area of the network into N parts with equal area. Each node is placed randomly in each divided part.

In each topology, we vary the node densities and the value of lattice parameter d. The node densities are varied among $5625m^2/node$, $10000m^2/node$, and $22500m^2/node$. d are varied among $100m$, $150m$, $200m$, $250m$, and $300m$.

Concerned with mobile networks, we use random way point mobility model (RWP) for evaluating HTL in an entity mobility model and reference point group mobility model (RPGM) for evaluating HTL in a group mobility model. We use IMPORTANT[15] to generate the mobile scenarios. Concerned with the scalability issue, we study the performance of HTL in both static and mobile network by varying the number of nodes.In static network, the uniform and random topology is chosen, when in mobile network, the RPW mobility model is chosen.

6.2 Results on Static Network

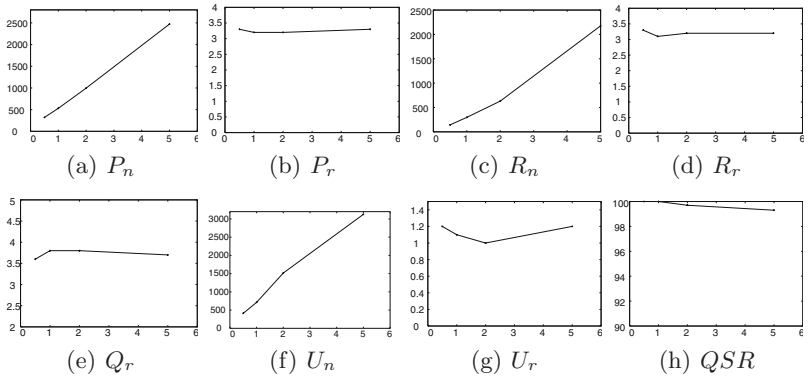
Table 3 gives the results of experiments on static networks.Based on the results shown above, we can verify the analytical results in the previous section. For example, in the case where the density of nodes is $10000m^2$ and d is $250m$, the distance between a node N and its virtual coordination $H(N.id)$ is among $[0, 1000\sqrt{2}]$. If the hash function distributes uniformity, then the average distance is about $700m$. The average store cost should be $700/250 \approx 3$, and the experiment result is 3.3. A little higher than theory result, but in a reasonable range. Concerned with the locality bounded property, in the random destination chosen model we used, the distance between the source and destination nodes is also approximately $700m$. Without locality, the cost for query should approximately equal to the cost of publish. In the result, the Q_r is smaller than P_r, which means that with locality, distances of a query packet travels decreases.

Table 3. Experiment results in static network

Density($m^2/node$)	$d(m)$	P_n	P_r	R_n	R_r	U_n	U_r	Q_r	LN_{min}	LN_{max}	LN_{ave}	QSR
5625	100	100	4.3	0	-	0	-	3.4	0	13	2.4	100%
5625	150	100	3.7	0	-	0	-	3.1	0	14	2.5	100%
5625	200	100	3.1	0	-	0	-	3.2	0	17	2.4	100%
10000	150	100	5.2	0	-	0	-	3.7	0	11	3.0	100%
10000	200	100	4.9	0	-	0	-	3.3	0	11	2.9	100%
10000	250	100	4.7	0	-	0	-	3.0	0	13	3.3	100%
22500	200	100	6.3	0	-	0	-	4.2	0	12	3.7	100%
22500	250	100	6.2	0	-	0	-	4.1	0	12	3.5	100%
22500	300	100	6.7	0	-	0	-	3.9	0	13	3.6	100%

6.3 Results on Mobile Network

Results on RWP: Figures 2(a), 2(c), 2(b), 2(d), 2(e), 2(f), 2(g), and 2(h) give the results of experiments on RWP. The conclusions can be drawn are:



(a) P_n (b) P_r (c) R_n (d) R_r

(e) Q_r (f) U_n (g) U_r (h) QSR

Fig. 2. Results of RWP

- The QSR decreases when the speed increases. This is also a problem in all current available location services as shown in [10] and [8]. In mobility situations, this occurs when the destination node is reachable, but the links to location servers are broken. Unlike other similar approaches, the location servers in HTL are not concentrated near the home region (or virtual coordination). Thus, the chances that all location servers are not reachable decreases, and the QSR is higher than other approaches.
- The results also show the locality property of HTL, Q_r is almost constant when the speed varies.

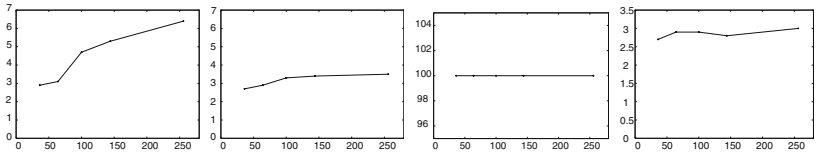
Results on RPGM: The experiment results on RPGM are given in table 4. The performance in RPGM is better than RWP in the case with comparable parameters, expecting that the P_r is higher in RPGM than that in RWP. The better performance is due to consistent of nodes movement in RPGM. The higher P_r is because in a group mobility mode, all nodes may be concentrated in a small area, where some virtual coordination are outside. Then a long routing path may be needed to find the corresponding location server.

Table 4. Experiment results on RPGM

P_n	P_r	R_n	R_r	U_n	U_r	Q_r	S_q	Loc_{max}	Loc_{ave}
1789	7.4	1543	6	2145	1.3	4.1	100%	29	4.7

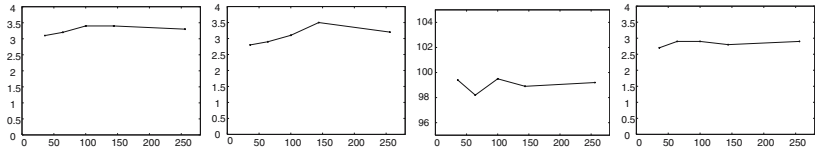
6.4 Results on Scalability

The results of the experiments on the issue of scalability of HTL are given in figure 3 and figure 4. Note that not all metrics defined in section 6.1 are used to evaluate the scalability of HTL. It is not only because the space limitation, but also because other metrics are not related to the issue of scalability.



(a) P_r (b) Q_r (c) S_q (d) Loc_{ave}

Fig. 3. Scalability of HTL in static networks



(a) P_r (b) Q_r (c) S_q (d) Loc_{ave}

Fig. 4. Scalability of HTL in mobile networks

7 Conclusion

In this paper we present a new location service named HTL, which possesses the property of locality bounded. The most remarkable of HTL is that HTL is the best one of so many location services proposed incorporating locality with minimal cost introduced. HTL employs a novel method to divide the physical space into lattices. The publish and query algorithms are based on this division. HTL is l^2 locality bounded, which means the cost of query is at most l^2, when the distance between the source and destination node is l. We give the analytical and simulation results in details in this paper.

Although HTL gives promising results both on theoretical analysis and simulative experiments. Some issues are worth future studying.

The first one is on the low bound of the locality bounded property. In both LLS and HTL, the result is $O(l^2)$. We doubt it is the best answer. If it is not, then what is the lower bound of locality a location service can achieve? Intuitionally, $O(l^2)$ may be the final answer. Think about a node who knows the destination is at most l away, but do not know the exact location. It should reach all possible locations, which is $O(l^2)$.

The second one is about HTL itself. Compared with other similar approaches, HTL needs more computing capability. In HTL, types of computing include computing on trigonometric function, manipulation on vectors and so on, all of which do not appear in other location services. Although the computing power of mobile devices are continuing increasing, reduce the complexity of HTL is still worth future investigating.

References

1. Karp, B.N.: Geographic routing for wireless networks. Ph.D. dissertation, Harvard Universtiy (2000)
2. Karp, B.N., Kung, H.T.: Greedy perimeter stateless routing for wireless networks. In: Proc. of the 6th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 2000), August 2000, pp. 253–254 (2000)
3. Bae, I.-H., Liu, H.: FHLS: Fuzzy Hierarchical Location Service for Mobile Ad Hoc Networks. In: Fuzzy Systems Conference FUZZ-IEEE (July 2007)
4. Chang, Y.-J., Shih, T.L.: Intersection location service and performance comparison of three location service algorithms for vehicular ad hoc networks in city environments. In: 3rd International Symposium on Wireless Pervasive Computing (May 2008)
5. Camp, T., Boleng, J., Wilcox, L.: Location Information Services in Mobile Ad Hoc Networks. In: Proc. of the IEEE International Conference on Communications (ICC) (2002)
6. Mauve, H.H.M., Widmer, J.: A survey on position-based routing in mobile ad hoc networks. IEEE Network (2001)
7. Li, J., Jannotti, J., Couto, D.S.J.D., Karger, D.R., Morris, R.: A scalable location service for geographic ad hoc routing. In: ACM MobiCom (August 2000)
8. Kasemann, M., Hartenstein, H., Fuber, H., Mauve, M.: Analysis of a location service for position-based routing in mobile ad hoc networks. In: Proc. of the 1st German Workshop on Mobile Ad-hoc Networking (WMAN 2002) (March 2002)
9. Abraham, I., Dolev, D., Malkhi, D.: Lls: a locality aware location service for mobile ad hoc networks. In: Proc. of the 2004 Joint Workshop on Foundations of Mobile Computing(DIALM-POM) (2004)
10. Saumitra, H.P., Das, M., Hu, Y.C.: Performance comparison of scalable location services for geographic ad hoc routing. In: 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2005) (2005)
11. Ratnasamy, S., Karp, B., Estrin, D., Govindan, R., Shenker, S.: Ght: A geographic hash-table for data-centric storage in sensornets. In: The First ACM International Workshop on Wireless Sensor Networks and Applications (WSNA 2002), Atlanta, GA (September 2002)
12. Liu, F., Rao, R.: Analysis of ght in mobile ad hoc network. In: The Proceeding of Third International Symposium on Parallel and Distributed Processing and Applications (ISPA 2005), Nanjing, China (November 2005)
13. Liu, F.: Research on location services in mobile ad hoc network, Masters thesis, Shanghai Jiaotong University (2005)
14. The network simulator - ns-2, `http://www.isi.edu/nsnam/ns/`
15. Helmy, F.B.N.S.A.: The important framework for analyzing the impact of mobility on performance of routing for ad hoc networks. Ad Hoc Networks Journal 1(4), 383–403 (2003)