uOttawa

L'Université canadienne
Canada's university

FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTDOCTORALES

u Ottawa

L'Université canadienne
Canada's university

FACULTY OF GRADUATE AND POSTDOCTORAL STUDIES

Jiulin Yang

AUTEUR DE LA THÈSE / AUTHOR OF THESIS

M.C.S.

GRADE / DEGREE

School of Information Technology and Engineering

FACULTÉ, ÉCOLE, DÉPARTEMENT / FACULTY, SCHOOL, DEPARTMENT

PMLS: A Position-maintained Location Service in Wireless Sensor Networks with Single Moving Actuator

TITRE DE LA THÈSE / TITLE OF THESIS

A. Nakak

DIRECTEUR (DIRECTRICE) DE LA THÈSE / THESIS SUPERVISOR

I. Stojmenovic

CO-DIRECTEUR (CO-DIRECTRICE) DE LA THÈSE / THESIS CO-SUPERVISOR

EXAMINATEURS (EXAMINATRICES) DE LA THÈSE / THESIS EXAMINERS

P. Flocchini

C.-H. Lung

Gary W. Slater

Le Doyen de la Faculté des études supérieures et postdoctorales / Dean of the Faculty of Graduate and Postdoctoral Studies

PMLS: A Position-Maintained Location Service in Wireless Sensor Networks with Single Moving Actuator

by

Jiulin Yang

A thesis submitted to
the Faculty of Graduate and Postdoctoral Studies
in partial fulfillment of
the requirements for the degree of

Master of Computer Science

Ottawa-Carleton Institute for Computer Science
University of Ottawa
Ottawa, Ontario, Canada

June 2009

Your file Votre référence
ISBN: 978-0-494-61356-6
Our file Notre référence
ISBN: 978-0-494-61356-6

Abstract

We consider the problem of updating locations of the actor with minimal message cost in Wireless Sensor Actuator Networks (WSAN) while still maintaining good hop counts between sensors and the actor. Existing solutions are flooding-based or rendezvous-based and involve high message cost. We introduce two fully-localized guaranteed-delivery low-message-cost location service algorithms for WSN with single slowly moving actor. One is called Position-Maintained Location Service for Random Movement (PMLS-RM) and the other is called Position-Maintained Location Service for Controllable Movement (PMLS-CM). Simulation results show that PMLS-RM outperforms Multipoint Relay variant of Doubling Circle (DC-MR) in networks with various node numbers and densities. It has dominant lead over DC-MR in low-degree or small size networks: over 70% reduction in message cost and same or better hop count dilation. Simulation results show that PMLS-CM also outperforms DC-MR in networks with various node numbers and densities. Similarly, PMLS-CM is superior to DC-MR in low-degree or small size networks: over 40% reduction in message cost and less than 1.4 hop count dilation in vast majority of the scenarios.

I hereby declare that I am the sole author of this thesis.

I authorize the University of Ottawa to lend this thesis to other institutions or individuals for the purpose of scholarly research.

Signature

I further authorize the University of Ottawa to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

Signature

Acknowledgements

I would like to acknowledge my sincerest gratitude to my supervisor Prof. Dr. Amiya Nayak and co-supervisor Prof. Dr. Ivan Stojmenovic at Ottawa-Carleton Institute for Computer Science. Without their enthusiastic and expert guidance during writing of this paper, it would not be possible to complete this thesis.

I would also like to thank my husband, for his consistent understanding and support in my whole stage of graduate study, and my father, Prof. Yang, for his valuable help in understanding the background and basic knowledge of Wireless Sensor Networks at the very beginning of my research.

Table of Contents

List of Figures

List of Tables

List of Acronyms

CDS	Connected Dominating Set
DC	Doubling Circle
DC-MR	Multipoint Relay variant of Doubling Circle
GFG	Greedy-Face-Greedy
GG	Gabriel Graph
MR	Multipoint Relay
PMLS	Position-Maintained Location Service
PMLS-CM	Position-Maintained Location Service for Controllable Movement
PMLS-RM	Position-Maintained Location Service for Random Movement
UDG	Unit Disk Graph
WSAN	Wireless Sensor Actuator Network
WSN	Wireless Sensor Network

List of Symbols

D	average node degree
E	ratio of sudden endpoint change to normal endpoint change
F	period of HELLO message exchange
L	level of message forwarding
N	node number
S	maximum moving speed
T	duration of simulation

Chapter 1 Introduction

1.1 Background and Motivation

A Wireless Sensor Network (WSN) is a wireless network consisting of spatially distributed autonomous devices, called *sensors*, which have integrated capabilities of sensing, processing and radio communication, and cooperatively monitor physical or environmental conditions, such as temperature, sound, vibration, pressure, motion, etc. Sensors have small size and use battery as their power supply, so they have limited energy, memory, computational speed and bandwidth.

A traditional WSN includes static sensors and a single static *sink*, which is directly connected to the user and collects information sent by sensors in a multi-hop way. In this thesis, we are more interested in a more generalized scenario, where there are several potentially-moving actuators besides a static sink. Such kind of WSN is called Wireless Sensor-Actuator Networks (WSAN). Actuators, also called *actor*s, are resource-rich devices and they create a high bandwidth backbone for communication between themselves and connection to the sink. In WSAN, sensors send their sensing data to their nearest actor, while the latter, based on the collected information, can make decision and perform appropriate actions on itself, sensors and/or the monitored environment.

Undoubtedly, routing problem is definitely one of the most important operations in WSN/WSAN. On the one hand, WSN usually consists of hundreds up to millions of tiny sensors, so the routing algorithm should be scalable. It has been experimentally confirmed [JPS, LJJCKM] that routing protocols that do not use geographic location in the routing decisions, are not scalable. Therefore, Stojmenovic [S-cm2002] points out that it is likely that only position-based routing algorithms, or *geographic* routing algorithms, provide satisfactory performance for large WSN. In this thesis, we only focus on geographic routing algorithms.

On the other hand, it is energy-consuming and thus impossible for sensors to learn and maintain the knowledge of the accurate topology of the whole network. Localized routing gives a solution to this problem. In a localized routing algorithm, each node makes a decision to which neighbor to forward the message based solely on the location of itself, its neighboring nodes and the destination, wishing the simple local behavior achieves the desired global objective. This is also called *fully-localized* algorithm. Clearly, localized routing algorithms are scalable.

In addition, *guaranteed delivery*, which refers to the ability of successfully forwarding a message from the source to the destination in a timely manner under the assumption of ideal MAC layer, is the primary goal of a routing task. Greedy-Face-Greedy (GFG) algorithm [BMSU, DSW] is loop-free geographic routing algorithm. Among all the neighbors closer to the destination than itself, current node forwards data packet to the node closest to the destination using greedy routing. If no such neighbors, face routing is applied until the packet reaches another node which is closer to the destination than the failure node, at which point greedy routing is resumed. Frey and Stojmenovic [FS] proved that GFG algorithm guarantees delivery in any planar graph. The experimental data [DSW] shows that enhanced by the dominating set concept in face mode, GFG algorithm compares very well with the global shortest path algorithm, especially for dense networks. In this thesis, we use GFG algorithm as our routing algorithm.

So far, we only discuss the routing problem in static network. When coming to the routing problem where the destination is moving, such as a moving actor in WSAN, things become much more difficult because the mobility of the destination causes frequent unpredictable topological changes. Generally, the routing problem in such case consists of two complementary tasks [LJS]:

(1) Location service (comprising of location update and location retrieval);
(2) Routing data packets from the source to the destination whose current location is known. If the moving speed of the destination is much smaller than the transmission speed, we can use any routing algorithm for static network, such as GFG, to finish the second task.

The first task is the heart of this thesis.

1.2 Problem Statement

Let us have the problem statement first. In this thesis, we only consider WSAN with single moving actor. The problem is about location service in WSAN; more precisely, it deals with *how to report new positions of the actor to sensors with minimal total number of messages so that routing from sensors to the actor still has good hop count.*

From the above statement, we can derive the following two objectives:

(1) The process of location update should be efficient, which is measured by its message cost (average number of location update messages generated or forwarded per node during one simulation);

(2) The result of location retrieval should be efficient too, which is measured by the hop count of the routing path from sensors to the actor.

To provide a good location service, we need to make tradeoff between overhead of location update and length of routing path.

1.3 Existing Solutions

Generally speaking, there are two ways to deal with the routing problem when the destination is moving. One is to route the long data packet directly toward the destination according to the current knowledge, and correct the routing direction on the road. The other is to get the exact location of the destination by sending short destination search tickets before routing long data packet to it, as described in three-phase routing scheme in Chapter 2. Intuitively, the former method has less cost if the location of the destination is accurate or almost accurate, while

the later method may have better performance if the location of the destination is inaccurate. Independent of the data routing, the moving destination needs to update its location from time to time. If location service only includes location update step through frequent network-wide flooding and no destination search step, it is *flooding-based* location service. If it contains both location update step and destination search step, it is called *rendezvous-based* location service.

Doubling Circle [APL] is a representative of the former type. The main idea of this scheme is to define a set of geographic routing zones, which are circles and each subsequent circle has twice larger radius than previous one, used for both location update and data routing. When the destination moves out of the scope of the i-th circle, it updates its new position to both i-th and (i+1)-th circles and defines a new i-th circle centered at current position of the actor. The data packet initiated by the source node follows the same circles of last update. It first routes toward the known position of the destination, then as the packet moves closer to the destination, it is adapted towards the center of the circle with twice smaller radius than the previous one, until the destination is eventually reached.

Amouris, Papavassiliou and Li [APL] proposed two variants of message forwarding. One is by flooding, the other is by intelligent flooding using Multipoint Relaying (MR) algorithm, called Multipoint Relay variant of Doubling Circle (DC-MR) in the thesis, where only MultiPoint Relays (MPRs) retransmit the message. MPR is 1-hop neighbor of the current node and the set of MPRs can cover all the 2-hop neighbors of the current node [SSW]. Since DC-MR can directly route data packets to the moving actor without destination step, which is similar with the algorithms proposed in the thesis, we use DC-MR as our competitive algorithm in the simulation.

Doubling Circle is certainly competitive, but when the destination is moving along one direction for a long time, the radius of largest circles may encompass almost all nodes of the network and the message cost of location update will be as high as flooding scheme. Moreover, the routing path discovered under this scenario may also be pretty long.

1.4 Assumptions and Limitations

As all the other algorithms, our proposed algorithms also contain assumptions and limitations. We assume the following:

- Each sensor uses omni-directional antennas so that it can receive a message from any of its neighbours.
- Sensors are fixed and distributed randomly in the monitoring field.
- Sensors are equipped with GPS or any other positioning service, so they are aware of their own positions.
- The whole sensor network is connected and placed in the Euclidian plane. It is modeled as Unit Disk Graph (UDG).
- There is only one actor in the network and it moves slowly. Actor always keeps connected to the sensor network while it is moving. At the very beginning, the actor broadcasts its initial position (in PMLS-RM) or its initial position and first endpoint (in PMLS-CM) by certain intelligent flooding.
- Actor is resource-rich and can receive and send messages using transmission radius $\geq R$. In this thesis, we assume its transmission radius is R, same as that of sensors.
- In the simulation, we assume ideal MAC and Physical layers without collisions.
- In the simulation, we use hop count as the distance metric.

1.5 Contributions

In this thesis, we introduce two fully-localized guaranteed-delivery location services, Position-Maintained Location Service for Random Movement (PMLS-RM) and Position-Maintained Location Service for Controllable Movement (PMLS-CM). They are for two types of mobility pattern respectively: one is called *random movement* and the other is called *controllable movement.*

Random movement refers to the pattern that the actor moves with uniform random speed and direction and it can change its direction and speed at any time. Controllable movement refers to the pattern that the actor moves to a known *endpoint*, the expected destination of the movement, with uniform random speed. The actor can change to a new endpoint with a new uniform random speed before arriving at the old endpoint. Such kind of changes may occur several times.

In the connected sensor network modeled as UDG graph, we prove that GFG can deliver the packet to the sensor node that is closest to a point (not a real node) among all sensors. Furthermore, we prove that a data packet routing to an out-of-date position of the actor (PMLS-RM) or the endpoint (PMLS-CM) can also make progress on its way to the destination. Base on this property, we prove that PMLS-RM guarantees delivery from any sensor to the mobile actor unless all the neighbours of the actor are disconnected between two successive location updates. We also prove that PMLS-CM guarantees delivery from any sensor to the mobile actor unless the actor is disconnected to all the sensors.

1.5.1 PMLS-RM

The basic idea of PMLS-RM is that the actor periodically exchanges HELLO message with its neighboring sensors. After each HELLO message exchange, whenever it finds at least one broken edge (the edge to actor doesn't exist any more due to disconnection with actor) or one new-made edge (the edge to actor is newly established due to connection with actor), the actor sends a LOCATION UPDATE message. If there is no broken node, the LOCATION UPDATE message only includes current position of the actor. If there is at least one broken node, but all the broken node(s) can be covered by a subset of the current 1-hop neighbors of the actor, called *recovery neighbors*, the actor will include id(s) of the recovery neighbor(s) in the LOCATION UPDATE as well. If there is at least one broken node which can not find corresponding recovery neighbor, the actor needs to send one RECOVERY LOCATION UPDATE message to each such broken node.

After receiving the LOCATION UPDATE message, current sensor applies GFG algorithm to find out the forwarding nodes for the old and new positions of the actor. If they are same, it keeps silent; otherwise, it retransmits.

After receiving the RECOVERY LOCATION UPDATE message, if it is not the destination (broken node), current sensor forwards the message. All the neighbors of the current sensor get the updated position of the actor by monitoring its retransmission. If it is the destination, current sensor initiates a LOCATION UPDATE message, containing current position of the actor only.

1.5.2 PMLS-CM

The basic idea of PMLS-CM is that after initial exchange of HELLO messages, based on its current neighbor list, the actor first calculates the minimal broken time and the corresponding broken node, and then sends a LOCATION UPDATE message to the broken node before it looses connection to the latter. Actor calculates next minimal broken time afterwards.

Similar with PMLS-RM, once receiving LOCATION UPDATE message, current sensor retransmits the message if using GFG algorithm, its forwarding nodes for the old and new endpoints are different; otherwise, it keeps silent.

Different from PMLS-RM, the actor can change its endpoint while moving. If endpoint changes, before moving towards the new endpoint, the actor sends a RECOVERY LOCATION UPDATE message to the old endpoint, including the location of new endpoint, new moving speed, etc. According to GFG algorithm, the message will advance toward old endpoint, and will eventually traverse the face containing it. The loop will be discovered on that face (repeating same edge twice), and further forwarding will be stopped. During the face traversal, all the nodes on the face enclosing the old endpoint get the updated information.

1.6 Simulation

We simulated our proposed algorithms with J-Sim v1.3 [J-Sim]. Our competitor is DC-MR. We make two groups of comparison: PMLS-RM vs. DC-MR and PMLS-CM vs. DC-MR. We measure the performance of the algorithms in three different aspects: *success rate* (ratio of number of packets sent to the actor to number of packets received by the actor), *message cost* (average number of location update messages generated or forwarded per node during one simulation) and *hop count dilation* (ratio of hop count of compared algorithm to hop count of Shortest Path algorithm). Our experiments were done on UDG graphs whose node number (N) ranges from 100 to 500 and average node degree (D) ranges from 8 to 24.

1.6.1 PMLS-RM vs. DC-MR

PMLS-RM shows apparently better performance than DC-MR in networks with various densities:

- In low-degree networks (D = 8), the message cost of PMLS-RM is 1.26, 2.62, 3.49 and 4.38 for N = 100, 200, 300 and 400 respectively and the reduction is from 79% to 45% for N = 100 to 400. Moreover, PMLS-RM can always reach 100% success rate while DC-MR cannot guarantee delivery for $N \geq 200$, and the hop count dilation of PMLS-RM is the same or better than that of DC-MR.
- In dense networks, by decreasing the level of message forwarding properly, the message cost of PMLS-RM decreases from 5.7 to 3.35 in the case of N = 300 and D = 24, 25% reduction to DC-MR's, and the hop count dilation of PMLS-RM is same as that of DC-MR.
- In normal-density networks (D = 14), the message cost of PMLS-RM is 1.31, 2.99, 4.11, 4.39 and 4.86 for N = 100, 200, 300, 400 and 500 respectively and the reduction is from 70% to 21% for N = 100 to 500, and the hop count dilation of PMLS-RM is a little bit bigger but very close to that of DC-MR.

Undoubtedly, PMLS-RM has dominant lead over DC-MR in low-degree networks or low-number networks: over 70% reduction in message cost, same or better hop count dilation and always 100% success rate while DC-MR cannot guarantee delivery for $N \geq 200$ when $D = 8$.

When the actor's moving speed increases, from 1m/s to 10m/s, PMLS-RM still has better performance than DC-MR: the reduction of message cost is from 30% to 50% for $N = 300$ and $D = 14$, and the hop count dilation changes between 1.1 and 1.5 while the hop count dilation of DC-MR varies between 1.1 and 1.4.

1.6.2 PMLS-CM vs. DC-MR

Regarding message cost, PMLS-CM also shows apparently better performance than DC-MR in networks with various node numbers and densities:

- In low-degree networks ($D = 8$), the message cost of PMLS-CM is 3.56, 4.44, 4.31 and 4.74 for $N = 100$, 200, 300 and 400 respectively and the reduction is from 40% to 48% for $N = 100$ to 400. Moreover, PMLS-CM can reach 100% success rate in all the scenarios except for $N = 400$, while DC-MR cannot guarantee delivery for $N \geq 200$. The hop count dilation of PMLS-CM varies between 1.5 and 1.9 while the dilation of DC-MR changes between 1.2 and 1.6.
- In dense networks, by decreasing the level of message forwarding properly, the message cost of PMLS-CM decreases from 4.82 to 3.67 in the case of $N = 300$ and $D = 24$, a 15% reduction from that of DC-MR. The hop count dilation of PMLS-CM is around 1.2 and the dilation of DC-MR is 1.1.
- In normal-density networks ($D = 14$), the message cost of PMLS-CM is 1.92, 3.65, 4.35, 4.06 and 4.13 for $N = 100$, 200, 300, 400 to 500 respectively and the reduction is from 62% to 32% for $N = 100$ to 500. The hop count dilation of PMLS-CM is less than 1.4 while the dilation of DC-MR is less than 1.2.

Similarly, PMLS-CM is obviously superior to DC-MR in low-degree networks or low-number networks: over 40% reduction in message cost, 100% success rate in all the scenarios except for $N = 400$ while DC-MR cannot guarantee delivery for $N \geq 200$.

The above simulation results were developed when the ratio of sudden endpoint change to normal endpoint change (E) is 1:2. When E gets bigger, from 1:2 to 1:1, PMLS-CM still has less message cost than DC-MR: the reduction is about 20% in the case of $N = 300$ and $D = 14$, and the hop count dilation is less than 1.5.

To conclude, our two algorithms, PMLS-RM and PMLS-CM, fulfill the two objectives of the above problem statement well and are indeed competitive to Doubling Circle scheme in various scenarios.

1.7 Thesis Organization

The remainder of the thesis is organized as follows. We give a literature review on the existing work on the problem of location service in WSN in Chapter 2. In Chapter 3 and 4, we elaborate PMLS-RM algorithm, followed by its performance analysis. In Chapter 5 and 6, we detail PMLS-CM algorithm, followed by its performance analysis. Finally, we conclude our work in Chapter 7 and discuss some future work as well.

Chapter 2 Literature Review

2.1 Geographic Routing

Geographic routing forms a specific class of routing protocols which requires that each node is able to determine its coordinates by means of a location system like GPS or relative positioning based on signal strength estimation. In WSN, it is usually executed in a localized manner, where each node only knows the position of itself, its 1-hop or 2-hop neighbors and the destination.

2.1.1 Greedy Routing

Greedy routing is a widely used routing principle among the localized geographic routing protocols. Each node always forwards messages to its neighbour which is closest to the destination based on some type of greedy path-finding heuristic. It is easy implemented and its hope count is very close to the shortest path algorithm whenever it is successful, especially in the dense networks.

Finn [F] proposed the greedy routing scheme based on geographic distance. Source node *S* selects neighbouring node which is closest to the destination among its neighbours. Only neighbours closer to the destination than *S* are considered. Otherwise, due to lack of advance, the method fails. Stojmenovic and Lin [SL] proposed a variant of this method called *Geographic Distance Routing* (GEDIR) scheme where all neighbours are considered, and the message is dropped if the best choice for a current node is to return the message to the node that the message came from.

2.1.2 Face Routing

The major drawback of greedy routing is high failure rate for sparse networks and it cannot guarantee delivery. This leads to the intensive study of planar graph routing, or *face routing*, which is generally used as a recovery mechanism for a greedy routing failure.

The basic idea of face routing is to planarize the network graph in a localized manner and forward a message along one or possibly a sequence of adjacent faces which provide progress towards the destination. Exploration of a single face can be done in a localized way by applying the well known *left-hand* or *right-hand* rule. Message forwarding according to the left-hand rule is similar to sending the message along the edge which is lying next in counter-clockwise direction from the previous visited edge. The right-hand rule in contrast sends the message to the edge lying next in clockwise direction.

2.1.3 Planar Subgraph Construction

Normally, an arbitrary wireless sensor network graph is not planar. Since a non-planar graph contains crossing edges which may leads to a routing loop during recovery, the planar subgraph is necessary for the recovery strategy to be loop-free. Thus, before any use of face routing, a planar graph construction mechanism has to be applied in advance.

Three prominent subgraph constructions, the *Gabriel Graph* (GG) [GS], the *Relative Neighborhood Graph* (RNG) [JT] and the *Localized Delaunay Triangulation* (LDT), have been proposed [GGHZZ, LCW, LSW].

[BMSU] proposed a algorithm to construct a planar connected subgraph, P, of the original Unit Disk Graph (UDG), G. P is the intersection of unit disk graph and Gabriel graph, GG, on the set S of N given nodes, defined as follows. Let $disk(u, v)$ be the disk with diameter (u, v). Then, the $GG(S)$ is a graph in which edge (u, v) is present if and only if $disk(u, v)$ contains no other points of S. It suffices to check, for each neighbor w of u, whether $|wm| < |uv|/2$, where m is the center of the disk. It was proved in [BMSU] that P is a connected planar graph and constructed in localized manner. We use this algorithm to planarize the generated UDG graphs in our simulation.

2.1.4 Greedy-Face-Greedy Routing

Bose, Morin, Stojmenovic and Urrutia [BMSU] proposed a GPS-based localized routing algorithm, Greedy-Face-Greedy (GFG), which is loop-free and guaranteed delivery for WSN modeled by unit disk graphs where nodes may communicate directly if their distance is smaller than the transmission radius *R*. This algorithm first constructs a planar subgraph, a Gabriel Graph, of the original unit disk graph in localized manner. Then it applies GEDIR algorithm until it either fails or reaches the destination. If GEDIR algorithm fails, face routing is applied until the packet reaches another node which is strictly closer to the destination than the failure node, at which point GEDIR algorithm is resumed.

DATTA, Stojmenovic and Wu [DSW] presented a concrete implementation of face routing enhance by shortcut procedure and routing via internal nodes. Performance analysis shows that GFG compares very well with the shortest path algorithm, especially for dense networks.

GFG algorithm was implemented by Karp and Kung [KK] with the consideration of realistic MAC layer and experiments with moving nodes. Their Greedy Perimeter Stateless Routing (GPSR) algorithm uses sparser RNG graph instead of its supergraph, GG graph, and changes faces before edge crossings instead of doing it afterwards. Frey and Stojmenovic [FS] proved that GPSR algorithm guarantees delivery only in RNG and GG graphs, but GFG algorithm [BMSU, DSW] guarantees delivery in any planar graph.

So we use GFG algorithm as our routing algorithm in this thesis.

2.2 Flooding-based Schemes

So far, a lot of algorithms have been proposed for the problem of location service.

Existing location services can be divided into two types: *Flooding-based* and *rendezvous-based.* Flooding-based schemes relies on flooding, which involves all or large portion of nodes in

the network, for location update. It usually only include location update step, no destination search step. Representatives of flooding-based protocols are Doubling Circle scheme and Request Zone scheme. Rendezvous-based schemes usually involve steps of both location update and destination search, which are part of the three-phase routing framework. Representatives of rendezvous-based schemes are Quorum-based scheme and Home-agent Based scheme. The former is guaranteed delivery which the latter is not.

2.2.1 Doubling Circle Scheme

Amouris, Papavassiliu and Li [APL] presented a position based multi-zone routing protocol for wide area mobile ad-hoc networks. In [APL], a set of geographic routing zones is defined for each node, which are circles and each subsequent circle has twice larger radius than previous one, used for both location update and destination search if applicable. When the destination moves out of the scope of the i-th circle, that is, the distance between the current position of the destination and the center of the i-th circle is greater than the radius of the i-th circle, the destination updates its new position to both i-th and $(i+1)$-th circles and defines a new i-th circle centered at its current position. [APL] proposed two variants to forward location update messages. One is by blind flooding (where every sensor receiving the message will forward it once, if inside the forwarding region). The other is by an intelligent flooding, where only nodes inside a backbone will forward the message. The backbone used is constructed by Multipoint Relaying (MPR) algorithm. Every node decides a set of 1-hop neighbors, called *MultiPoint Relay* (MPR), which cover all the 2-hop neighbors. Upon receiving the message, node first checks if it is MPR node of the sender. If so, it will retransmit, and will add to the packet the list of its MPR nodes [SSW].

The data routing process initiated by the source follows the same circles of last update. It firstly routes data to the currently known position of the destination, and as the data moves closer to the destination, it adapts the data towards the center of the circle with twice smaller radius than the previous one, until the destination is eventually reached.

This process is illustrated in Fig. 2.1, source node *S* wants to send data packet to the destination *D*. Since *S* is outside the 2nd circle and inside the 3rd circle of *D*, *S* sends packet to the center of the 3rd circle, which is position *D*3 in Fig. 2.1. As soon as the packet steps into the 2nd circle, it is redirected toward the center of the 2nd circle, which is position *D*2. Similarly, as the packet enters the 1st circle, it is redirected toward the center of the 1st circle, which is position *D*1. Eventually, the packet is forwarded to the exact position *D*.

Performance analysis shows that Doubling Circle scheme is a scalable bandwidth-efficient routing mechanism. Its drawback is that the radius of larger circles may encompass almost all nodes of the network if the actor continues to move toward same direction for a long time and the routing paths discovered in such scenario are pretty long, which make the scheme not applicable for the applications with high service quality requirements. A similar algorithm, using squares instead of circles, and additional sophisticated techniques, is proposed in [LJCKM].

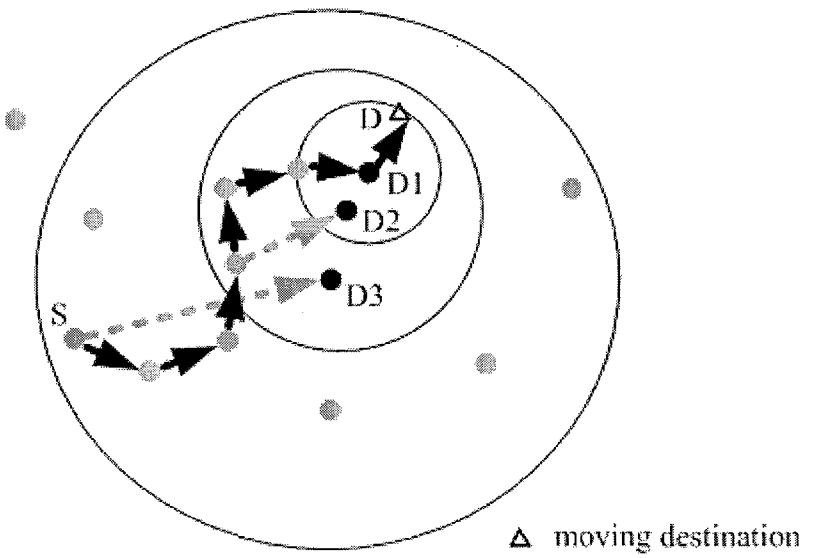

Fig. 2.1. Doubling Circle: routing from *S* to *D*

2.2.2 Request Zone Scheme

A Distance Routing Effect Algorithm for Mobility (DREAM) is described in [BCSW]. The source or any intermediate node *A* calculates the direction of the destination *D* and, based on the mobility information about *D*, chooses an angular range, called *request zone* in [KV], which is determined by the tangents from *A* to the circle centered at *D* with radius equal to a maximal possible movement of *D* since the last location update. The message is forwarded to all neighbors whose direction belongs to the selected range. The DREAM algorithm incorporates the idea of triggering the sending of location updates by moving nodes autonomously at a rate and hop distance that correspond to the node's mobility rate.

The definition of the request zone was modified in [SRL]. Stojmenovic, Ruhil and Lobiyal [SRL] proposed a unified framework for both routing and geocasting problems, where geocasting problem is solved by routing towards the center of the geocasting region and that center is not necessarily needed to be a node of the network. Based on this unified framework, [SRL] proposed three methods, *R-DIR*, *V-GEDIR* and *CH-MFR*. in which the message is forwarded to exactly those neighbors which may be the best choices for a possible position of the destination. In *R-DIR*, these neighbors may include one or two neighbors that are outside of angular range if they have the closest direction to the circle. In *V-GEDIR*, these neighbors are determined by the concept of Voronoi diagram among the set of neighbors closer to the destination than current node. In *CH-MFR*, the concept of convex hull is analogously used to choose the candidates. These three methods all concentrate on the performance outside the geocasting region in case of routing. If the geocasting region is not connected itself, the nodes that are reached may be disconnected from the destination and thus message delivery is not guaranteed.

Stojmenovic [S-geo] addressed this problem by proposing three geocasting algorithms to guarantee delivery. Two of them are face traversal approaches and are based on depth-first search of the face tree and traversal of all faces that intersect the border of geocasting region, respectively. In the third approach, entrance zone multicasting based approach, the monitoring

center divides entrance ring of geocasting region into zones of diameter which equals to the transmission radius. Then the problem is decomposed into multicasting toward center of each zone, and flooding from these nodes.

For all the request zone algorithms, the destination is assumed to be located inside a circle or square, centered at its latest known location, whose size depends on available mobility information. So this scheme works pretty well when the destination is moving in uniform rectilinear motion, but may miss the destination in the estimated request zone if its moving speed changes often.

2.3 Rendezvous-based Schemes

2.3.1 Three-phase Routing Framework

When the destination is moving, the most difficult part of the geographic routing is the inaccuracy of the position of the destination. Routing the full data to the estimated location of the destination leads to high communication overhead if it misses the destination there. So Stojmenovic [S-q1999] suggested to divide the routing problem into several components and to study each of them separately. More precisely, all the message traffic related to routing tasks is divided into four components:

(1) *Location update message*s are initiated by the moving destination, which acts on its movement.

(2) *Destination search message*s are initiated by a source node when it wants to route a data packet to the destination. The source node usually does not use blind flooding as a means of search, but an intelligent search strategy differed in various location service schemes.

(3) *Path creation message*s are initiated by the destination upon receiving the first copy of a destination search message. The destination is able to find the best path to the source according to the accurate position of the source. Since the transmission speed is far greater than node

movement speed in the majority of the scenarios, the geographic routing algorithms for static networks may be applied for the path creation phase.

(4) *Data traffic message*s are initiated by the source node upon receiving reply from the destination containing its exact location, possibly together with the path toward it. The source may, alternatively, attempt to create another path toward the location of the destination by applying any geographic routing algorithms for static networks.

In [S-q1999], messages are divided into short and long ones. Short messages do not have the real data, and therefore has much lesser number of bits than the long message that contains the real data. Location update, destination search and path creation messages are all short messages, which are called *control message*s in this thesis.

Since location update messages are generated independently on routing request, we generally regard above routing framework as *three-phase routing framework.* In this framework, destination search and path creation messages are generated by routing requests.

Some schemes are not presented according to this three-phase routing framework, but they can be tailed to it. For example, the geocasting algorithms in [SRL] may be used for the destination search phase instead of sending the 'long' data message. Some schemes are designed under this general routing framework, such as the Quorum-based scheme and Home-agent Based scheme below.

2.3.2 Quorum-based Scheme

Quorum-based approaches for information dissemination are based on replicating information at multiple nodes acting as repositories. Such a query-and-update strategy has been previously employed for location management in cellular networks. Karumanchi, Muralidharan and Prakash [KMP] discussed information dissemination in partitionable mobile ad hoc networks. They replicated information at multiple nodes acting as repositories and employed quorum-based strategies to update and query information. In [KMP], n nodes are divided into $n^{1/2}$ groups with $n^{1/2}$ nodes in each group and preserve such quorums while nodes move. They also discussed the

question when to update location, and concluded experimentally that the best strategy is to update when a certain pre-specified number of links incident on a node have been established or broken since the last update, which is adopted in this thesis.

The main problem of [KMP] is that quorums are themselves fixed, and the movement of nodes can make nodes in the same quorum far apart from each other. So visiting all the nodes in the same quorum becomes a difficult task sometimes. Stojmenovic [S-q1999] proposed a different quorum-based strategy to deal with network dynamic. In [S-q1999], every moving node uses a counter to count the number of edge changes. When the counter reaches a fixed threshold value, the moving node forwards its new location information (and its identifier) within a '*column*', that is, to the north and south of its current location with certain 'thickness' of reporting. The destination search initiated by the source then begins with two tickets being sent within a '*row*', that is, to the east and west direction with certain 'thickness'. When the tickets reach each end of current 'row', the search is continued toward best reported destination's location, with corrections along the path as better information becomes available closer to the destination. A third ticket can be sent from the source directly toward the destination, to take advantage of possibly correct information on destination's location.

The advantage of [S-q1999] is that nodes do not stay in the fixed 'column' and the updated location information of the moving destination is easier to disperse. The frequent problem with this strategy is that the northernmost or southernmost node determined by the 'column' update may only be locally northernmost or southernmost. The 'horizontal' destination search can miss such a node when the latter remains above/below the 'row'.

This problem is overcome in [SLJ]. Each locally northernmost node may switch to FACE mode until another node, more northern, is found. It then converts back to regular upward move. This switch can be repeated few times. The process terminates when the first encountered edge of the outer face is about to be repeated. The final result is that all nodes on the outer face receive the location update. The search for the destination is performed similarly, using horizontal east-west direction instead. This method guarantees that 'horizontal' destination search and 'vertical'

location update will intersect at one or more nodes on the outer face. The resulting drawback is that nodes on the outer face will have more traffic demands. All the four variants considered are applicable in scenarios when all nodes move out of the original region, but preserving connectivity. They also adapt well to synchronous node movement (such as vehicles on a highway), where nodes keep mutual distances but move at high speed.

2.3.3 Home-agent Based Scheme

Stojmenovic [S-ha] proposed a Home-agent Based scheme for mobile ad hoc or sensor networks which is similar to the one used in cellular phone networks and mobile IP. In this scheme, at the beginning, each node informs every other node about its initial position, which will be its *home agent*. More precisely, home agent of a node consists of all nodes that are currently located inside a circle with radius pR (p is a network parameter), centered at the initial position of the node. Each moving node uses a counter to count the number of edge changes. When the counter reaches a fixed threshold value, it sends a location update message to its home agent using GEDIR or GFG algorithm. Nodes on the path and their neighbors also update information about this moving node. Similar with Quorum-based scheme, the source node sends two destination search tickets. One is send toward the destination; the other is sent toward the center of the home agent circle of the destination and stops at the first encountered home agent. This home agent will inform all the other agents the most recent location information about the destination and collected the replies from them. It then redirects the ticket toward the latest location of the destination collected so far and probably corrections its direction along the path.

Home-agent based scheme does not forward location update message by flooding, so it can provide high success rate with reasonable communication overhead. However, it does have some drawbacks. A common problem of this scheme is that sometimes if the destination is far away from home agent circle, the message cost for location update is high and the path from the source to the destination is long. In some scenarios, such as rescue missions or military actions, the destination may even moves out of the region where all home agents are located. In such

scenario, all agents are ineffective, and new home agent needs to be created. If such kind of movement is intensive, this scheme becomes ineffective. Some repair techniques, such as designing new home agent upon certain number of inefficient destination searches, can be applied, but their effectiveness is also limited to a 'small' movement.

Chapter 3 Position-Maintained Location Service for Random Movement

As we know, the mobility pattern of the actor has various types. Our Position-Maintained Location Service (PMLS) focuses on two of them: one is called *random movement* and the other is called *controllable movement.*

Random movement refers to the pattern that the actor moves with random speed and direction and it can change its direction and speed at any time. An example of random movement is that the actor is carried by an animal which moves aimlessly and unpredictably.

Controllable movement refers to the pattern that the actor moves to a known *endpoint*, the expected endpoint of the movement, with uniform random speed. An example of controllable movement is that when some emergent event happens, such as failure of a sensor, the actor moves to that spot to deal with it. If some event with higher priority happens before actor reaches the endpoint, the actor has to change its endpoint and handle the new event first. In this thesis, we adopt this more realistic controllable movement. That is, the actor moves to a known endpoint with some uniform speed and it may change to a new endpoint with a new uniform random speed during the movement. Such kind of changes may occur several times.

Apparently, the location services for these two mobility patterns are different. So, we propose two variants of PMLS in the thesis, one for each mobility pattern, called Position-Maintained Location Service for Random Movement (PMLS-RM) and Position-Maintained Location Service for Controllable Movement (PMLS-CM) respectively.

3.1 An Example First

The basic idea of PMLS-RM is that the actor periodically exchanges HELLO message with its neighboring sensors. After each HELLO message exchange, whenever it finds a broken edge or a new-made edge, the actor sends location update message to its neighboring sensors.

Upon receiving the location update message, the sensor compares the forwarding nodes for the old and new positions of the actor using GFG algorithm. If they are same, it does not retransmit the message; otherwise, it retransmits. The broken node receives the updated location of the actor either via the retransmitted location update message from its neighbor, or via the recovery location update message initiated by the actor.

Let us look at an example first. In Figure 3.1, the actor moves from location A_1 to A_2. At location A_1, a, b, u are its neighboring sensors; at location A_2, b, v are its neighboring sensors. So, the nodes a and u are broken nodes while node v is a new neighbor. At location A_2, actor sends location update message to its current neighbors, b and v.

For new neighbor v, because its forwarding nodes for the old and new positions of the actor are different, it always retransmits the location update message.

For broken node, the focus is how to report the updated location of the actor to it so that routing from the broken node to the actor will succeed. There are two situations.

(a) The broken node is the neighbor of one of the current neighboring nodes of the actor, like broken node a, which is the neighbor of b. In this case, b will retransmit the location update message even if its routing path to the actor does not change.

(b) The broken node is not the neighbor of any of the current neighbors of the actor, like node u, actor needs to send a recovery location update message to u using GFG algorithm. Based on 1-hop positional information or 2-hop topological information, the actor is easily able to detect such a broken node because it is not within its current two-hop neighborhood. In Figure 3.1, this recovery location update message follows the path *A2->b->a->c->d->e->f->g->h->i->j->w->u*. And all the nodes on this routing path get the current location of the actor.

Both nodes a and u will send the new location of the actor to its neighbors because the forwarding nodes for the old and new positions of the actor are always different for broken nodes.

For the other nodes, which are neither new neighbors nor broken nodes, like node c and w, if their forwarding nodes for the old and new positions of the actor are different, they

retransmit the message; otherwise, they keep silent. So *c* retransmits because its forwarding node changes from node *a* to node *b*, while *w* does not.

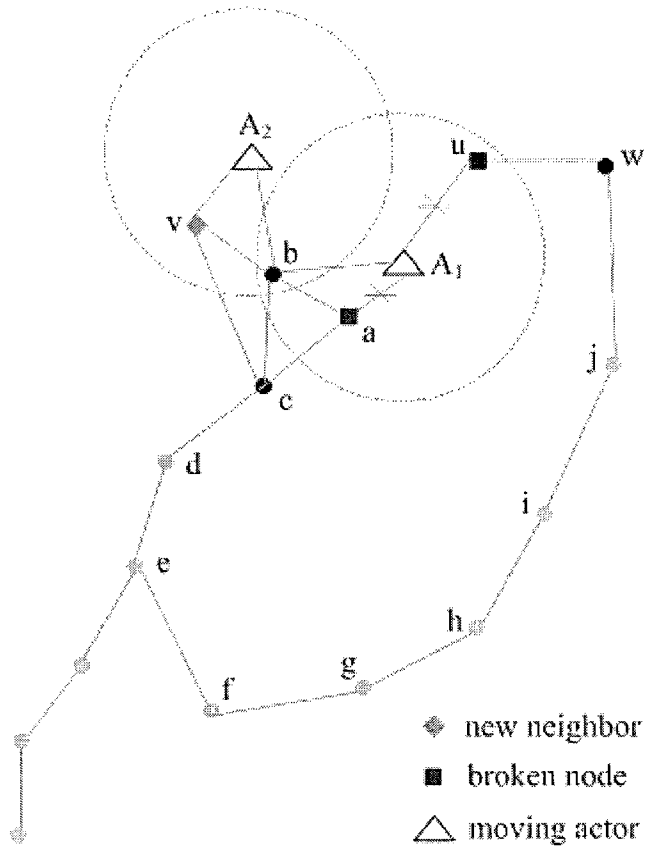

Fig. 3.1. PMLS-RM: an example

3.2 Description of Algorithm

The corresponding algorithm, PMLS-RM, is described as follows:

1. The actor and its neighboring sensors periodically exchange HELLO messages.
2. Each time after exchanging HELLO messages, the actor compares the new neighbor list with the old neighbor list and finds out which are the broken nodes and which are the new neighbors.

3. If there is at least one broken node or one new neighbor, the actor will send a LOCATION UPDATE message to its current neighbors. At the same time, the actor needs to ensure that all the broken nodes and new neighbors will successfully receive its new location.
   (a) For new neighbors, the actor just adds them to the neighbor list.
   (b) For broken nodes, in addition to removing them from the neighbor list, the actor also applies the following strategy:
      (i) The actor puts all the broken nodes into an *uncovered list*. It first finds the neighbor, which has the maximal number of broken nodes as its neighbor(s). Such kind of neighbor is called *recovery neighbor* and is put into *recovery neighbor list*. All the broken nodes it covers are removed from the uncovered list. This process repeats until the *uncovered list* is empty or there is no neighbor can cover any left node in the *uncovered list*.
      (ii) If both *recovery neighbor list* and *uncovered list* are empty, the actor sends a LOCATION UPDATE message to all its neighbors.
      (iii) If *recovery neighbor list* is not empty, the actor sends a LOCATION UPDATE message to its neighbors with designated recovery neighbors.
      (iv) If *uncovered list* is not empty, there are two choices here:
         - For each of these uncovered broken nodes, say u, the actor sends a RECOVERY LOCATION UPDATE message regarding u as the destination using GFG algorithm. So there may be several unicast messages in the network. In this thesis, we implement this option.
         - The actor sends one multicast RECOVERY LOCATION UPDATE message regarding all the uncovered broken nodes as destinations. We need to apply GFG variant for multicast in this case [SRLS].
4. When a sensor receives the LOCATION UPDATE message, it records the new position of actor first, then

(a) If it is the new neighbor of the actor, it adds the actor to its neighbor list and retransmits the message.

(b) If it is the broken node, it removes the actor from its neighbor list and retransmits the message.

(c) If it is the recovery neighbor included in the LOCATION UPDATE message, it retransmits the message even if according to GFG algorithm, the forwarding nodes for the old and new positions of the actor are the same.

(d) If it does not belong to any of the above three types, but the forwarding nodes for the old and new positions of the actor are different, it retransmits the message; otherwise, it keeps silent.

5. When a sensor receives the RECOVERY LOCATION UPDATE message, it records the new position of the actor too, then

(a) If it is not the destination (or one of the destinations) of the RECOVERY LOCATION UPDATE message, it forwards the message using GFG algorithm for unicasting or GFG variant for multicasting, depending on the type of the receiving message, a unicast message or a multicast message. All the neighbors of this sensor get the updated position of the actor by monitoring its retransmission.

(b) If it is the destination (or one of the destinations) of the RECOVERY LOCATION UPDATE message, it initiates a LOCATION UPDATE message to all its neighbors and its neighbors retransmit the message if their forwarding nodes for the old and new positions of the actor change.

3.3 Pseudo Code

We list the algorithm for the actor part and for the sensor part separately.

Algorithm 1-a: PMLS-RM (actor part), actor A, unicasting version

1: Find out new neighbors and broken nodes after exchanging HELLO message; Put broken nodes into *uncovered list*
2: **Repeat**
3: Find the neighbor of A, say w, who covers maximal number of broken nodes in *uncovered list*; Remove covered nodes from *uncovered list*; Put w into *recovery neighbor list*
4: **UNTIL** *uncovered list* = ∅ **OR** all neighbors of A are considered
5: **if** *uncovered list* = ∅ **AND** *recovery neighbor list* = ∅ **then**
6: A sends LOCATION UPDATE message to all neighbors
7: **if** *recovery neighbor list* ≠ ∅ **then**
8: A sends LOCATION UPDATE message with designated recovery neighbors to all neighbors
9: **if** *uncovered list* ≠ ∅ **then**
10: A sends a RECOVERY LOCATION UPDATE message to each broken node in *uncovered list*

Algorithm 1-b: PMLS-RM (sensor part), sensor u

1: **while** *true* **do**
2: check the type of message m received
3: **switch** *type* of message **do**
4: **case** *type* = 'LOCATION UPDATE'
5: **if** u is new neighbor **then** add A to neighbor list and retransmit
6: **else if** u is broken node **then** move A from neighbor list and retransmit
7: **else if** u is recovery neighbor **then** retransmit
8: **else if** forwarding nodes for old and new positions of A are different **then** retransmit
9: **case** *type* = 'RECOVERY LOCATION UPDATE'
10: **if** u is not destination **then**
11: forward message according to GFG algorithm and all its neighbors get the updated position of actor by monitoring u's retransmission
12: **else if** forwarding nodes for old and new positions of A are different **then**
13: u initiates LOCATION UPDATE message to all neighbors

3.4 Proof of Guaranteed Delivery

In all the following proofs, we assume that the sensor network is connected and whenever the actor sends location update message, at least one neighbor of last update can receive it. In other words, the value of period of HELLO message exchange is small enough so that at least one neighbor is connected to the actor between two successive location updates.

Lemma 1. *Suppose that destination for GFG routing is a point X that may not be a sensor node, GFG (based on GG) will deliver the packet to the sensor node N that is closest to X among all sensors.*

Proof. If X is a sensor node, GFG guarantees delivery to X according to [FS]. If X is not a sensor node, GFG will eventually terminate on a loop containing X inside, and loop is recognized by repeating the initial node along the same edge [FS]. It remains to show that the closest sensor to X is also on that face. This is true if Gabriel Graph is used in creating faces. Suppose that, by contradiction, the closest sensor N to endpoint X is not on the GG face containing X (note that X is not part of GG). Then XN intersects one edge of the face, say AB, and since N is the closest node, $|NX| < |AX|$, $|NX| < |BX|$. Then $\angle NAB < \angle NAX < \angle ANX$. Similarly, $\angle NBA < \angle NBX < \angle BNX$. Thus $\angle NAB + \angle NBA < \angle ANX + \angle BNX = \angle ANB$, meaning that the $\angle ANB$ is obtuse. But this then means that AB is not in GG, which is a contradiction, as illustrated in Figure 3.2. And the nodes on that face can learn which one is closest to X during the traversal. Also, during the face traversal, all the nodes on the face get the updated information about the actor. □

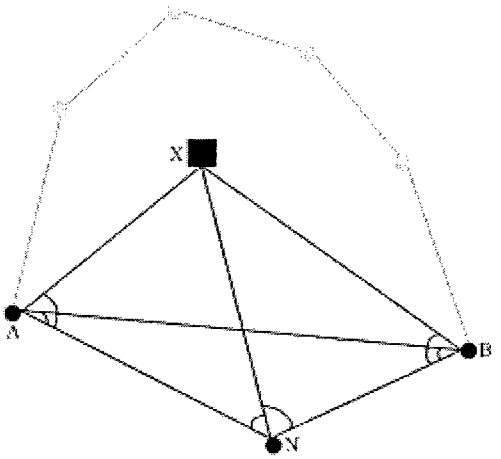

Fig. 3.2. PMLS-RM: proof of Lemma 1

Lemma 2. *Suppose the actor performs location update at positions* $A_1, A_2, \ldots$, *the initial position is* A_0 *and current position is* A_k. *In PMLS-RM applied on a connected sensor network, a sensor with the knowledge of* A_i ($i < k$) *can always route its data packets to a sensor with the knowledge of* A_j ($i < j \leq k$).

Proof. When a sensor u with the knowledge of A_i, that is, u knows the position of A_i, routes the data packets to the actor using GFG algorithm, routing succeeds if the actor is currently at A_i. In this case, from Lemma 1, the closest node to position A_i, say w, will receive the packets. From assumptions, we know that A is always connected to at least one sensor in the network, so w must be a neighbor of A. When actor A moves to A_{i+1} and sends location update there, three cases exist:

(a) w is still neighbor of A_{i+1}, as illustrated in Figure 2.2(a). In such case, w will get the knowledge of A_{i+1} directly from the actor.

(b) w is not a neighbor of A_{i+1}, but has a neighbor who is the neighbor of A_{i+1}. In such case, w will get the knowledge of A_{i+1} from this neighbor, node v in Figure 2.2(b).

(c) w is neither a neighbor of A_{i+1}, nor has a neighbor who is the neighbor of A_{i+1}, as illustrated in Figure 2.2(c). In such case, w will get the knowledge of A_{i+1} via RECOVERY LOCATION UPDATE message.

In any of the above three cases, w will has the knowledge of A_{i+1} finally.

During the routing towards position A_i, before arriving at node w, u may also encounter a node who has the knowledge of A_j ($j > i$). In summary, u can always route its packets to a node who has the knowledge of A_j ($i < j \leq k$). □

Theorem 1. *PMLS-RM algorithm provides guaranteed delivery to routing from any sensor to the slowly mobile actor for any connected sensor network modeled as unit disk graph, and actor movements with endpoints within communication range of at least one sensor.*

Proof. Suppose actor A performs location updates at positions of A_1, A_2, ... and its initial position is A_0 and current position is A_k. From assumptions, every sensor knows initial position A_0 of A. Suppose sensor u with knowledge of A_0 sends a packet to A. Based on Lemma 2, during its routing to A_0, it always can encounter a node, say w_i, who has the knowledge of A_i ($0 < i \leq k$). Thus the advance toward A_k is made, and iteratively further steps can also be made. That is, if $i < k$, during w_i's routing to A_i, w_i will encounter a node who has the knowledge of A_x ($i < x \leq k$). Eventually, the packet will be forwarded to a node w_k that has the knowledge of A_k which forwards the packet to actor A, as illustrated in Figure 2.3. □

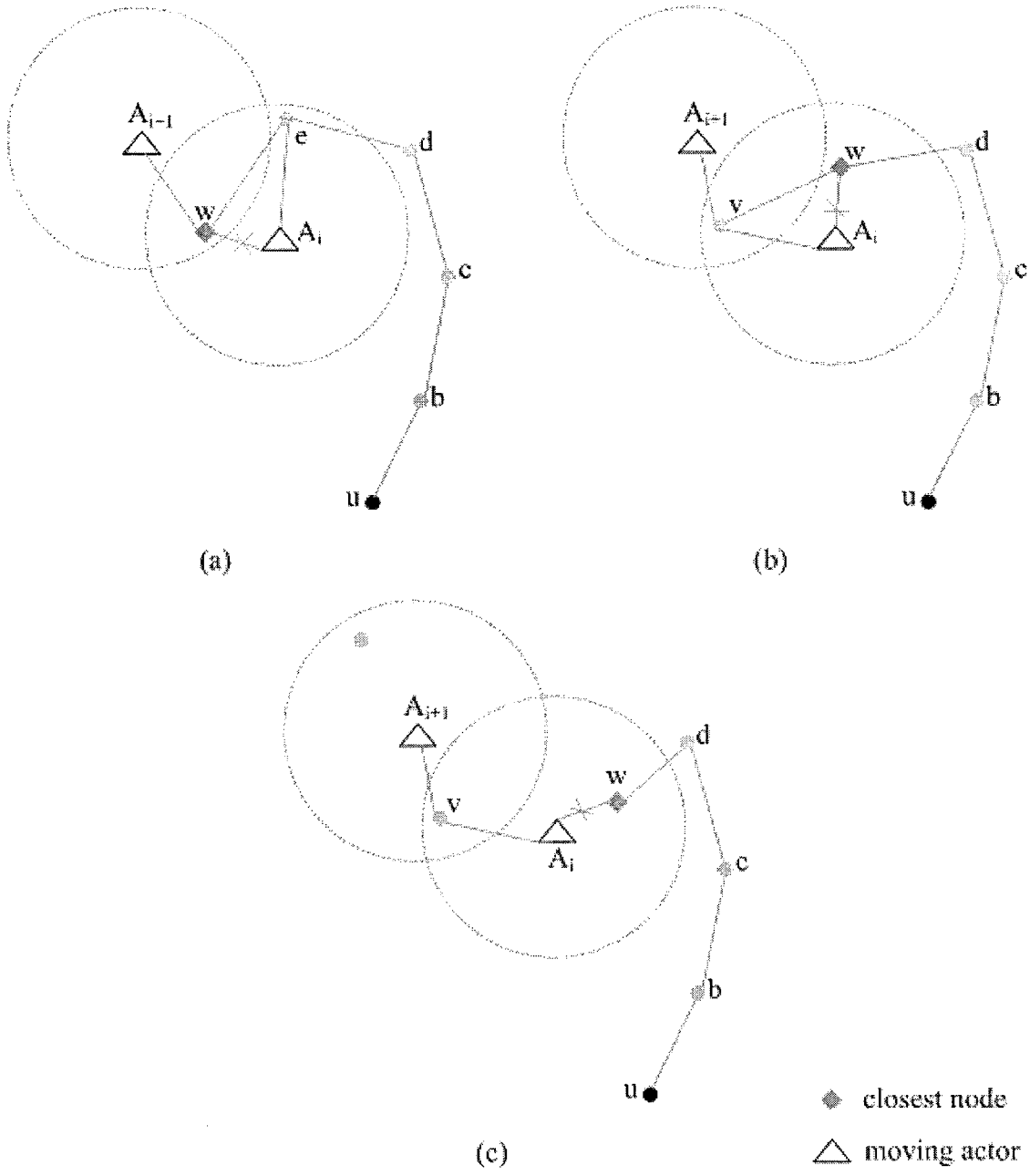

Fig. 3.3. PMLS-RM: three cases of routing to position A_i. (a) w is 1-hop neighbor of A_{i+1}. (b) w is 2-hop neighbor of A_{i+1}. (c) w is neither 1-hop neighbor nor 2-hop neighbor of A_{i+1}.

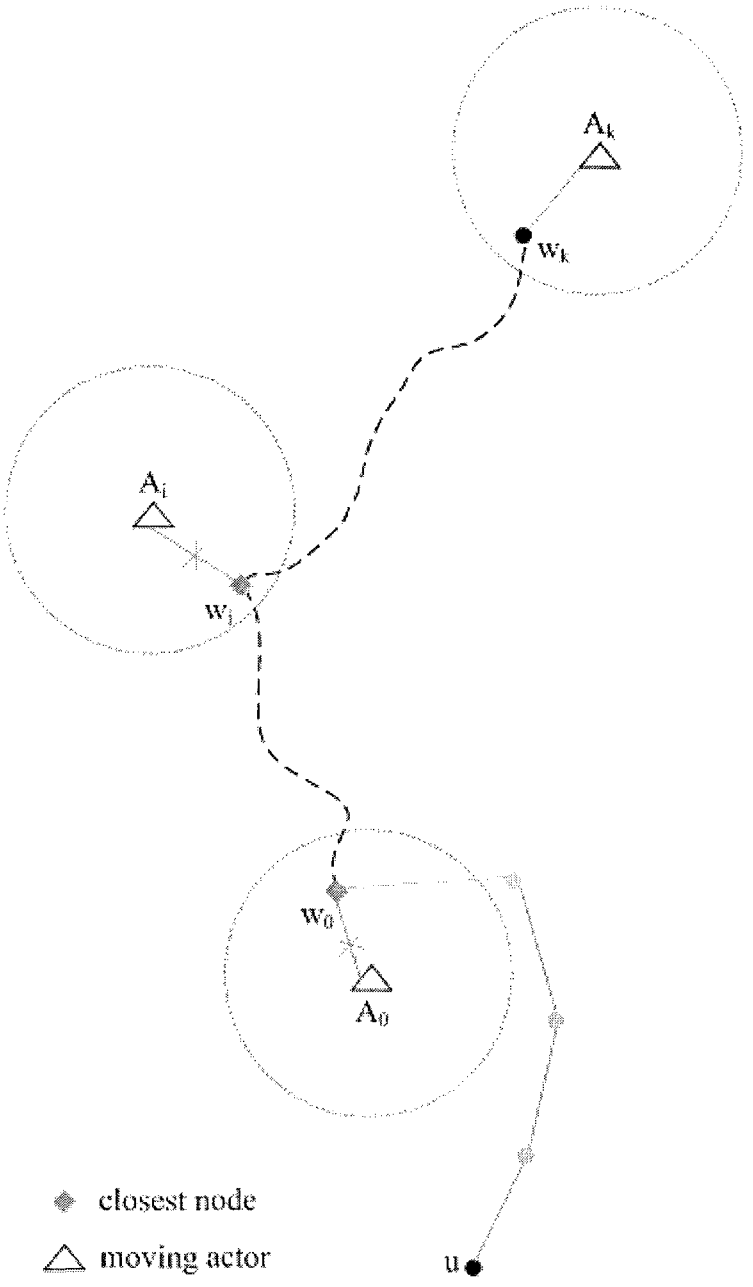

Fig. 3.4. PMLS-RM: guaranteed delivery in connected sensor network modeled as UDG graph.

Chapter 4 Performance Evaluation of PMLS-RM

Our simulations compare PMLS-RM with Doubling Circle (DC) scheme. The reason for choosing DC for comparison is that DC does not necessarily follow the three-phase routing principle. In other words, DC can directly route data packets to the moving actor, or used as a method for destination search step under three-phase routing framework if needed. PMLS-RM has the same character. As we mentioned before, DC has two variants: one is by flooding and the other is by Multipoint Relay (MR). Since the message cost of flooding is much bigger, we use only MR variant in this thesis. Moreover, for the purpose of comparison, we use GFG algorithm as the routing algorithm for both PMLS-RM and DC-MR.

4.1 Simulation Setup and Parameters

The simulator we used is J-Sim v1.3 [J-Sim]. The size of monitoring area is 100m × 100m. The experiments were carried using random unit disk graphs, which were generated in a standard way as follows. Each of n sensors (not including the actor) is chosen by selecting its x and y coordinates at random in the interval [0, 100*)*. We sort all $n(n-1)/2$ (potential) edges in the network by their length, in increasing order. The radius R that corresponds to the chosen value of average node degree d is equal to the length of $(nd/2)$-th edge in the sorted order. Dijkstra's shortest path algorithm (from one source to all other nodes) is used to test whether a graph is connected. Generated graphs which are not connected are ignored. Initially, the actor is at position [50, 50], which is connected to all the testing UDG graphs.

In order to evaluate algorithms' performance in different scenarios, we simulated different network environments by alternating the following parameters:

- N (number of sensors): In order to test the impact of node number on the performance, we vary value of N among 100, 200, 300, 400 and 500.

- D (average node degree): In order to test the impact of average node degree on the performance, we vary valued of D among 8, 10, 12, 14, 16, 18, 20, 22 and 24.
- S (maximum moving speed): The mobility model we used is *random waypoint model*. The actor moves to a randomly chosen destination position on a straight line with a uniform speed that is also chosen at random. Once the actor reaches this destination, it will not stop but immediately randomly choose a new destination position plus a new moving speed and continue its movement. Since actor is moving slowly, majority of the experiments were done with S equal to 1m/s, but impact of higher speed is also discussed.
- T (duration of simulation): The duration of each simulation is 1000s, where each of 100 randomly selected source nodes sends one data packet to the moving actor at random time. The final result is the average of 10 simulations based on 10 different underlying UDG graphs respectively. The UDG graphs, source nodes and their start time of routing are same for the compared algorithms.
- F (period of HELLO message exchange): The period of HELLO message exchange has influence on the performance of PMLS-RM. Unless clearly stated, we set $F = 9$s, but bigger and smaller F are also discussed.
- L (level of message forwarding): In the algorithm description of PMLS-RM in Chapter 3, we supposed that location update messages are forwarded as far as possible unless the forwarding nodes for the old and new positions of the actor are same. During the simulation, we found that if we do not restrict the level of message forwarding, the message cost of PMLS-RM is pretty big although the hop count dilation is very close to shortest path algorithm. Since small message cost is the focus of this thesis, unless clearly stated, we restrict $L = a$ for $N = a \times 100$. That is, $L = 3$ when $N = 300$. Smaller L is also discussed.

4.2 Performance Metrics

To assess the performance of compared algorithms, we considered the following performance metrics.

- *Success rate*: ratio of number of packets sent to the actor to number of packets received by the actor. If guaranteed delivery, success rate is 1.
- *Message cost*: average number of location update messages generated or forwarded per node during one simulation (1000s). Although different nodes may have different traffic, for example, the neighbors of actor usually have heavier traffic, this metric still clearly indicates the local traffic of a sensor.
- *Hop count dilation*: ratio of hop count of the compared algorithm, PMLS-RM or DC-MR, to hop count of the Shortest Path (SP) algorithm. This metric provides an indication of how much the algorithm diverges from the SP algorithm.

4.3 Simulation Results

4.3.1 Message Cost

In this section, we compare the message cost between PMLS-RM and DC-MR in terms of average node degree D and node number N respectively.

Figure 4.1 shows that, when $N = 300$, the message cost of PMLS-RM increases while D increases, but the message cost of DC-MR decreases while D increases. For small D, such as $D = 8$, the message cost of PMLS-RM is only around 50% of that of DC-MR. The reason is that in low-degree networks, the set of Multipoint Relays used in DC-MR is almost the same as the set of 1-hop neighbors, which means almost every node in the network needs to forward the location update message once.

With the increase in D, PMLS-RM's lead over DC-MR gets smaller. Finally, DC-MR defeats PMLS-RM in dense networks. The reason is that one Multipoint Relay can cover more 2-

hop neighbors now, thus the set of Multipoint Relay gets smaller. As a result, the number of nodes needed to forward the location update message in DC-MR decreases when D increases. On the contrary, in PMLS-RM, for same value of L, more nodes are involved in message forwarding, so the message cost of PMLS-RM continually increases when D increases and eventually exceeds that of DC-MR.

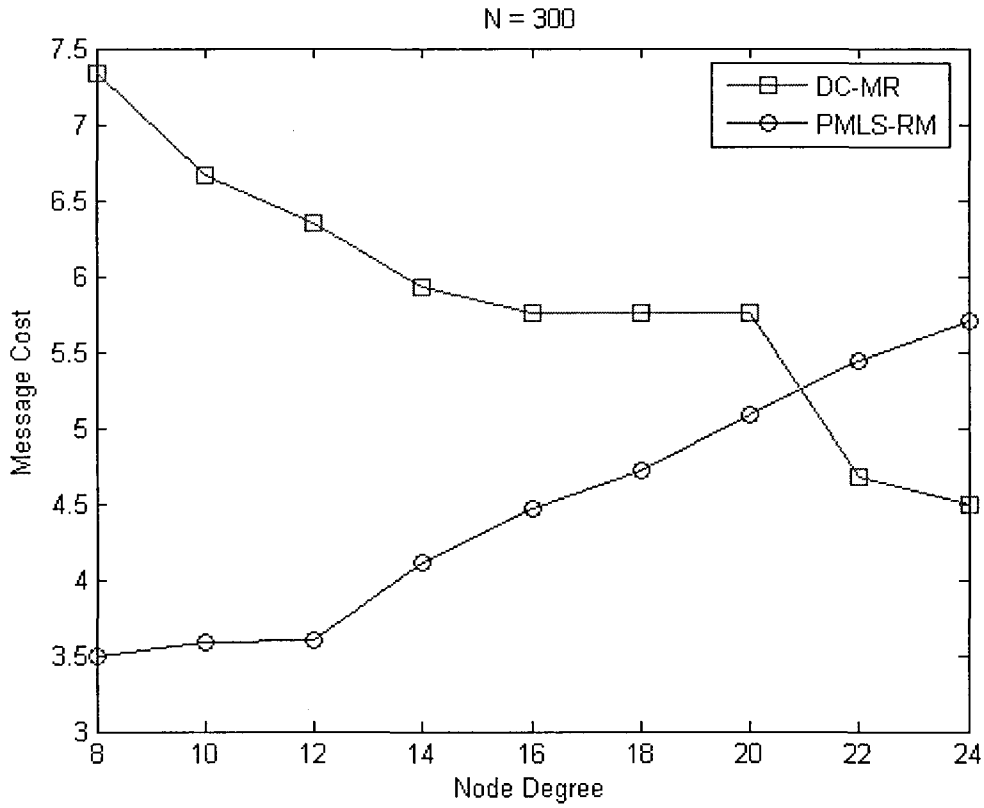

Fig. 4.1. PMLS-RM vs. DC-MR: message cost at various node degrees

Figure 4.2 shows that, when D = 14, the message cost of both PMLS-RM and DC-MR increases while N increases. In all the cases, PMLS-RM is superior to DC-MR, 70% to 20% reduction for N = 100 to 500. The reason why PMLS-RM dramatically outperforms DC-MR in low-number networks, such as N = 100, is that in such scenario, almost all the 1-hop neighbors are in the set of Multipoint Relay, which leads to the high message cost of DC-MR. However, when N gets bigger, the size of set of Multipoint Relay gets smaller, which offsets the impact of

increasing N, so the increasing speed of DC-MR is slower than that of PMLS-RM, leading to the smaller difference between them.

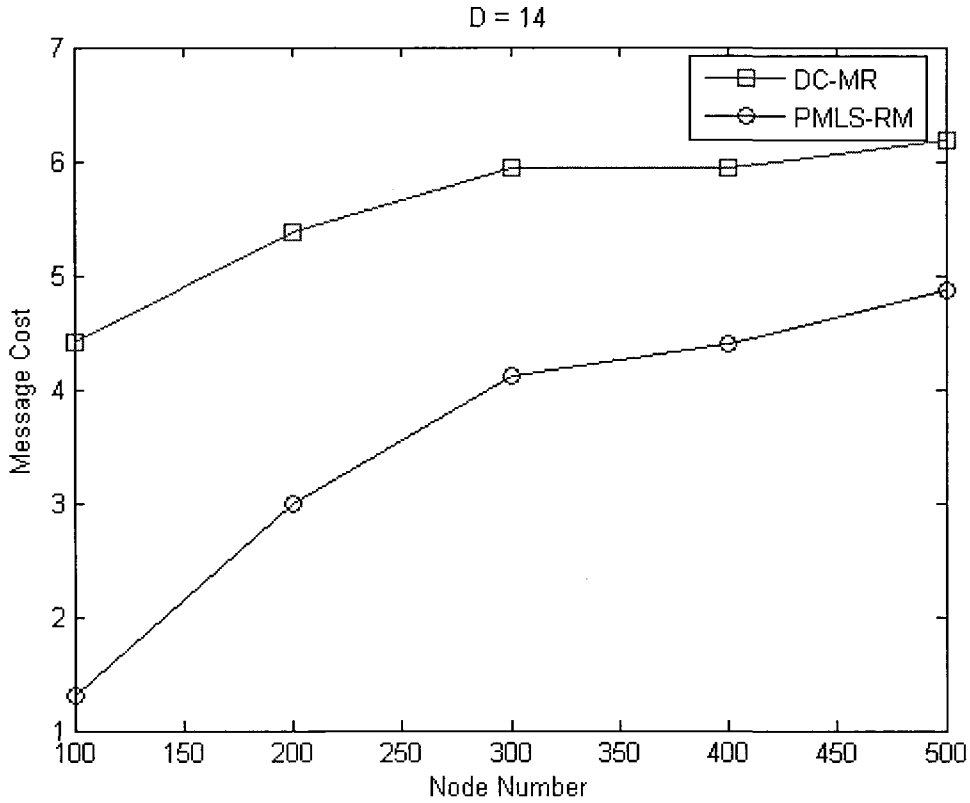

Fig. 4.2. PMLS-RM vs. DC-MR: message cost at various node numbers

In summary, except in dense networks, PMLS-RM shows apparently better performance than DC-MR in message cost. PMLS-RM's lead over DC-MR is quite large in either low-degree networks or low-number networks. We will elaborate these two scenarios later.

4.3.2 Hop Count Dilation

In this section, we compare the hop count dilation between PMLS-RM and DC-MR in terms of average node degree D and node number N respectively.

Figure 4.3 shows that, when $N = 300$, the hop count dilation of both DC-MR and PMLS-RM decreases while D increases because in dense networks, the performance of GFG algorithm

is very close to SP algorithm [DSW]. Except few cases when D is small, the difference between PMLS-RM and DC-MR is small and close to zero in dense networks. In low-degree networks, not strange at all, PMLS-RM has better performance than DC-MR in hop count dilation too.

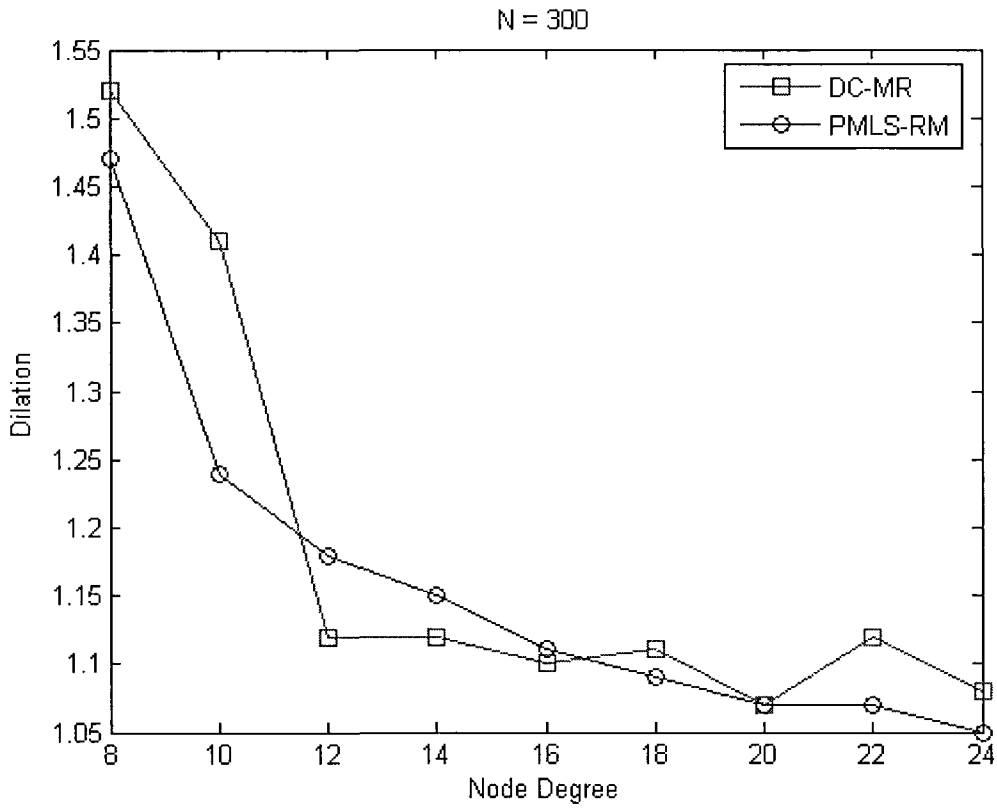

Fig. 4.3. PMLS-RM vs. DC-MR: hop count dilation at various node degrees

Figure 4.4 shows that, when D = 14, the hop count dilation of DC-MR remains stable while the hop count dilation of PMLS-RM changes slightly with the increase of N. The reason is that the set of Multipoint Relay depends more on node degree than on node number. In all cases, the difference between DC-MR and PMLS-RM is very small: PMLS-RM is slightly better than DC-MR when N is small while DC-MR is slightly better than PMLS-RM when N gets big.

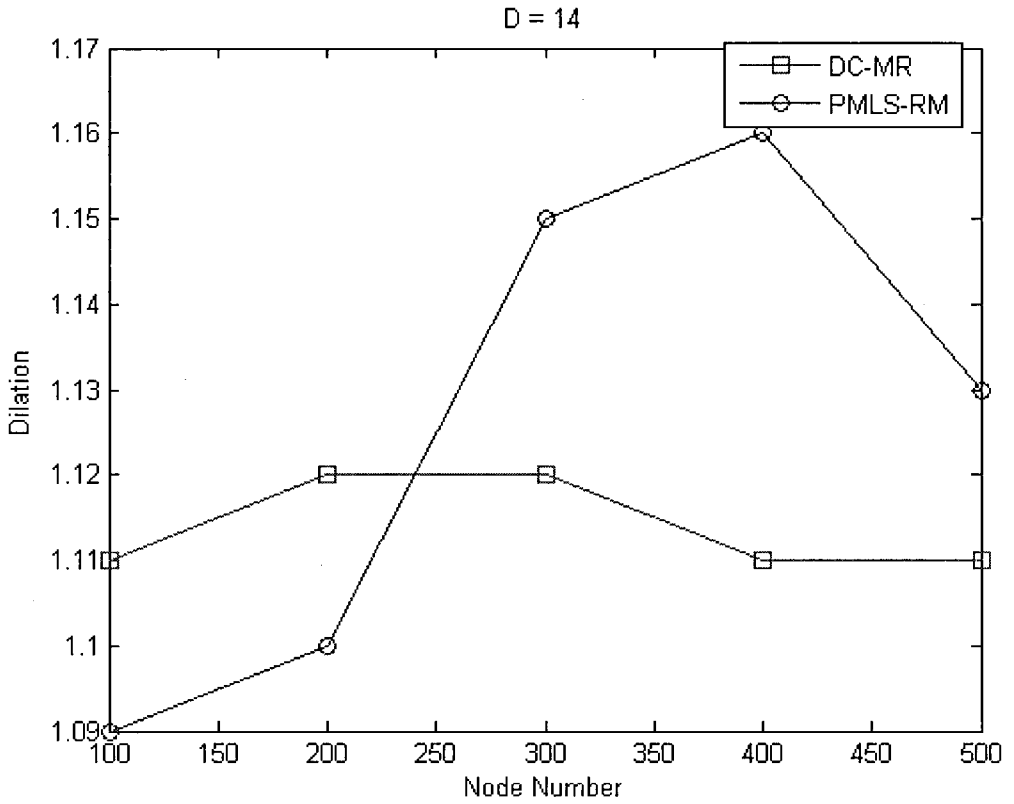

Fig. 4.4. PMLS-RM vs. DC-MR: dilation at various node numbers

In summary, the difference between PMLS-RM and DC-MR in hop count dilation is small. In low-degree or low-number networks, PMLS-RM is slightly better than DC-MR; in high dense networks, their difference is approaching zero; in the other cases, DC-MR is slightly better than PMLS-RM.

4.3.3 Impact of Moving Speed

In this section, we study the impact of moving speed (S) on both message cost and hop count dilation of the compared algorithms. We compare four values of maximum speed: 1m/s, 2m/s, 5m/s and 10m/s. We fix $N = 300$ and $D = 14$ to focus more on the impact of moving speed and reduce the affect of node number and average degree.

Figure 4.5 shows that, when $N = 300$ and $D = 14$, the message cost of both DC-MR and PMLS-RM increases when S increases. In all the cases, PMLS-RM is superior to DC-MR and the difference between them increases while the speed increases, 50% reduction for $S = 10$m/s. The reason is that with high moving speed, the actor has more opportunities to move outside certain circle, thus leading to more frequent location updates. In contrast, due to the fixed period of HELLO message exchange, the period of location update in PMLS-RM is fixed, so increase of moving speed only results in more forwarding nodes, which is the same for DC-MR.

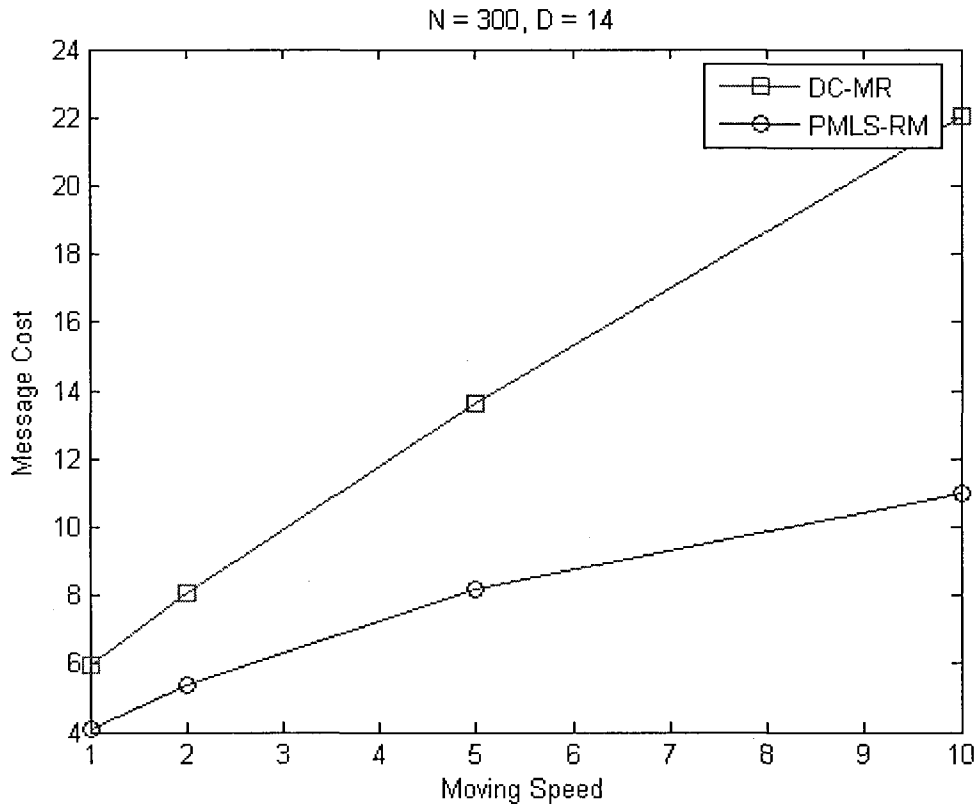

Fig. 4.5. PMLS-RM vs. DC-MR: message cost at various moving speeds

Figure 4.6 shows that, when $N = 300$ and $D = 14$, the hop count dilation of both DC-MR and PMLS-RM increase while S increases. In all the cases, DC-MR is superior to PMLS-RM because there are more nodes getting the location update messages in DC-MR than in PMLS-RM, thus leading to better routing paths.

When S = 10m/s, both DC-MR and PMLS-RM are not guaranteed delivery, so only successful deliveries, which are less than 100, are counted. So the hop count dilation of both DC-MR and PMLS-RM decreases in this case.

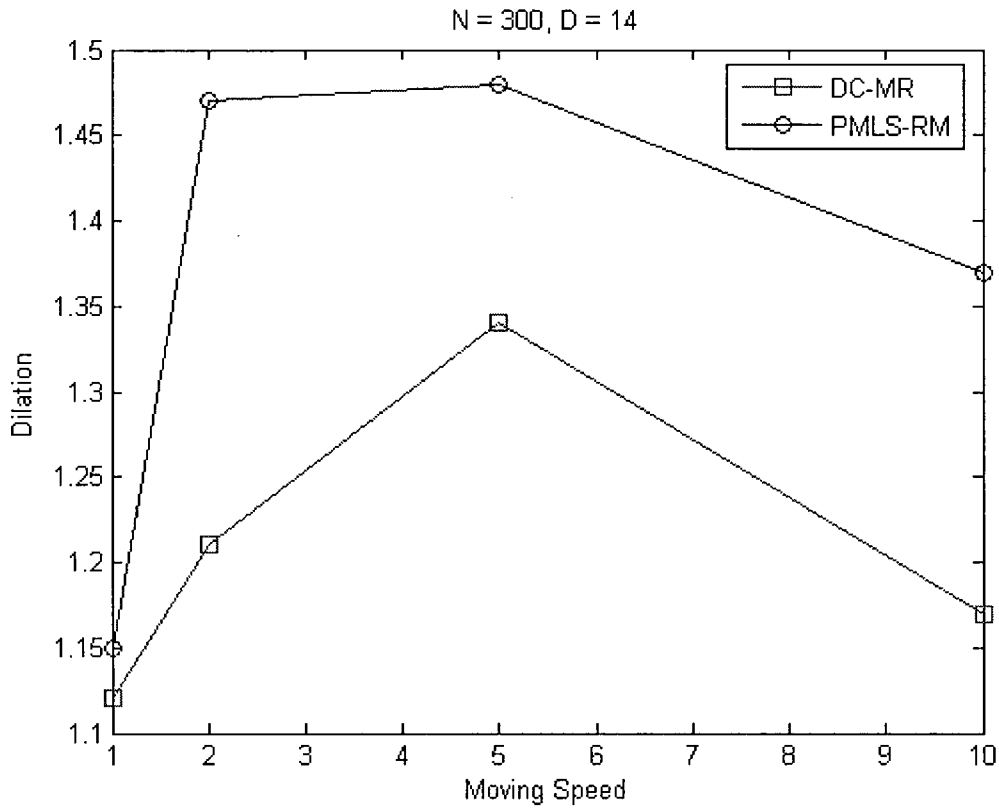

Fig. 4.6. PMLS-RM vs. DC-MR: dilation at various moving speeds

The reason PMLS-RM cannot guarantee delivery in the scenario of high speed is that all the neighbors of last location update are disconnected before next location update, so the routing tasks generated between two successive location updates will miss the actor. To solve this problem, we change the period of HELLO message exchange from 9s to 5s. As illustrated in Table 4.1, after adjustment, the message cost has over 20% growth (still over 30% reduction when compared with DC-MR), but the hop count dilation decreases besides guaranteed delivery.

$N = 300$, $D = 14$, $S = 10$m/s	Success rate	Message cost	Hop count dilation
DC-MR	99.4%	22.03	1.17
PMLS-RM ($F = 9$s)	96.4%	11.01	1.37
PMLS-RM ($F = 5$s)	100%	13.73	1.26

Table 4.1. PMLS-RM: Decrease F to guarantee delivery when $S = 10$m/s

In summary, PMLS-RM has better performance than DC-MR when moving speed of the actor increases; in particular, the message cost is only 50% of that of DC-MR. When the speed is high, both PMLS-RM and DC-MR cannot guarantee delivery, but PMLS-RM can reach 100% delivery rate if the period of HELLO message exchange is decreased properly.

4.3.4 Performance in Low-degree Networks

In this section, we study the performance of DC-MR and PMLS-RM in low-degree networks. We set $D = 8$ and vary N from 100 to 400.

Figure 4.7 shows that, when $D = 8$, the message cost of both DC-MR and PMLS-RM increase while N increases. In all the cases, PMLS-RM is notably better than DC-MR, around 80% to over 45% reduction for N = 100 to 400. The difference between them decreases with the increase in the number of nodes. The reason PMLS-RM has dominant lead over DC-MR in low-degree and sparse networks (80% reduction for $N = 100$ and $D = 8$) is that in such cases, the set of Multipoint Relay is almost the same as the 1-hop neighbor set, so the performance of DC-MR has same message complexity as the flooding. Conversely, since the level of message forwarding is only 1 for $N = 100$, only very small portion of nodes will retransmit the location update messages in PMLS-RM.

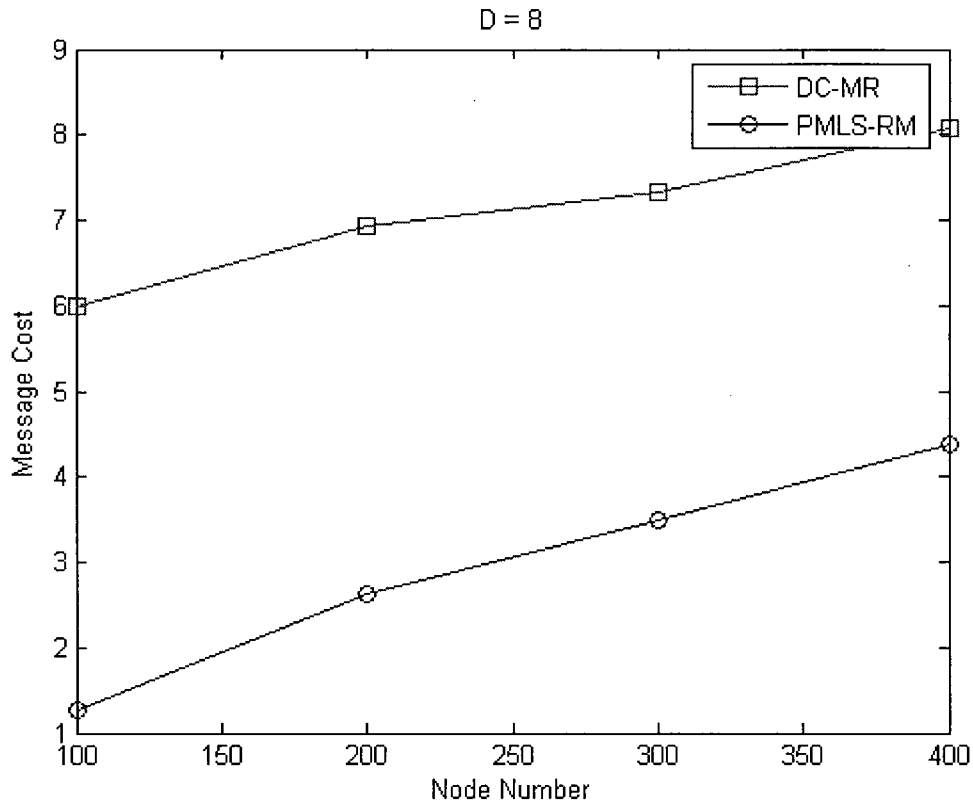

Fig. 4.7. PMLS-RM vs. DC-MR: message cost in low-degree networks

Figure 4.8 shows that, when $D = 8$, the hop count dilation of both DC-MR and PMLS-RM increases while N increases. The difference between them increases with the increase in N as well. In sparse networks, DC-MR is slightly superior to PMLS-RM because the number of nodes getting the location update messages in DC-MR is bigger than that in PMLS-RM. When N increases, PMLS-RM becomes superior to DC-MR. The reason is that in low-degree networks, the position of the source node has more impact on the hop count dilation and success rate of the compared algorithms.

Take Figure 4.9 as an example. $C1$, $C2$ and $C3$ are three location update circles and they all centered at location D initially. Since source node S is located in $C3$, so it sends data packet to the center of circle $C3$, which is location D. Suppose when the packet reaches location D through node w, actor moves to location D', which is still inside circle $C1$. When node degree is low, it is

more likely that all the neighbors of the actor at location *D* are disconnected to it when it moves to *D*', thus none of actor's previous neighbors will know actor's latest location *D*'. Furthermore, the actor itself cannot receive any data packet through its previous neighbors at location *D*, but can only receive the data packet through its current neighbors at location *D*'. In other words, when node number increases, the success of delivery relies more on the position of source nodes. That is why success rate of DC-MR decreases when *N* increases, as illustrated in Figure 4.10. Even if the data delivery is successful, the routing path will be longer. The same thing happens to PMLS-RM as well. To guarantee delivery, we decrease *F* from 9s to 6s. Table 4.2 is the results. Data in Figure 4.8 and Figure 4.9 when N = 400 are also the data with *F* = 6s.

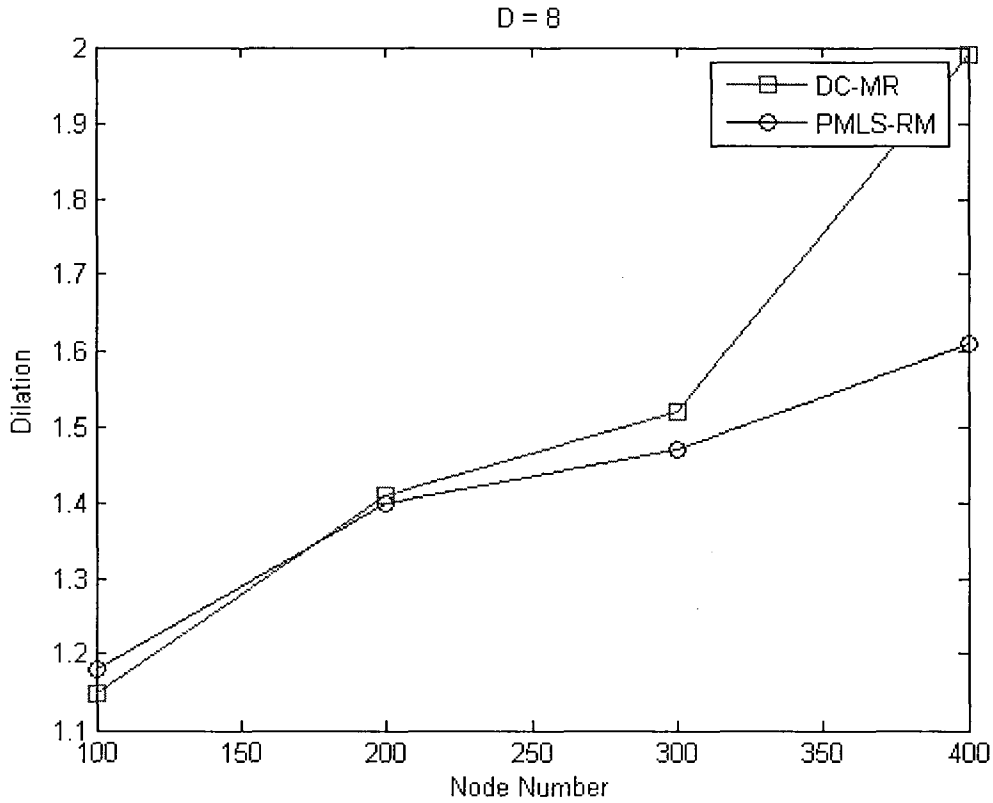

Fig. 4.8. PMLS-RM vs. DC-MR: dilation in low-degree networks

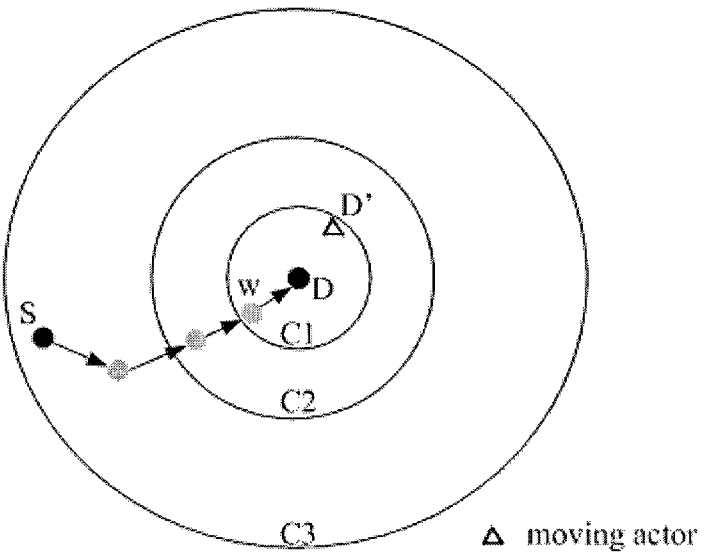

Fig. 4.9. DC-MR: no guaranteed delivery in low-degree networks

$N = 400$, $D = 8$, $S = 1$m/s	Success rate	Message cost	Hop count dilation
DC-MR	97.9%	8.08	1.99
PMLS-RM ($F = 6$s)	100%	4.38	1.61

Table 4.2. PMLS-RM: decrease F to guarantee delivery when $N = 400$ and $D = 8$

In summary, in low-degree networks, PMLS-RM greatly outperforms DC-MR in every compared metric. When the number of nodes is big, we can decrease the period of HELLO message exchange to guarantee delivery at the cost of slight increase of message cost and hop count dilation.

4.3.5 Performance in Dense Networks

In this section, we study the performance of DC-MR and PMLS-RM in dense networks.

As illustrated in Figure 4.1 and Figure 4.2, DC-MR performs better than PMLS-RM in dense networks. Here we only use the case of $N = 300$ and $D = 24$ as the example and discuss two ways to improve the performance of PMLS-RM in such scenario.

One intuitive way is to increase the period of HELLO message exchange since we already see its impact on the PMLS-RM's performance in the previous sections. Table 4.3 gives the comparison results, for both F = 9s and F = 20s. When F increases, the message cost of PMLS-RM decreases as expected but is still bigger than that of DC-MR. At the same time, the hop count dilation of PMLS-RM slightly increases but still smaller than that of DC-MR as well. Since to guarantee delivery, the decrease of F is not unlimited, this method is not an effective solution.

N = 300, D = 24, S = 1m/s	Message Cost	Hop count dilation
DC-MR	4.49	1.08
PMLS-RM (F = 9s)	5.7	1.05
PMLS-RM (F = 20s)	5.13	1.06

Table 4.3. PMLS-RM: increase F to reduce message cost in dense networks

The second method is to decrease the level of message forwarding (L), from 3 to 2. The motivation is that we think in dense networks, such kind of change won't have any bad impact on success rate or hop count dilation except reducing message cost. Table 4.4 proves our assessment. The decrease of L significantly cuts the message cost of PMLS-RM (25% reduction from that of DC-MR) with no change of guaranteed delivery and slight increase of hop count dilation.

N = 300, D = 24, S = 1m/s	Message Cost	Hop count dilation
DC-MR	4.49	1.08
PMLS-RM (L = 3)	5.7	1.05
PMLS-RM (L = 2)	3.35	1.08

Table 4.4. PMLS-RM: decrease L to reduce message cost in dense networks

In summary, in dense networks, the performance of DC-MR is generally better than that of PMLS-RM. However, we can still manage to make PMLS-RM superior to DC-MR by decreasing the level of message forwarding properly.

4.3.6 Impact of Period of HELLO Message Exchange

In this section, we briefly summarize the impact of period of HELLO message exchange on all the compared metrics.

Generally, smaller F will increase both message cost and success rate but decrease hop count dilation meanwhile. On the contrary, bigger F will decrease message cost but increase hop count dilation at the same time. If the value of F is too bit so that all the neighbors of last update are disconnected to the actor before next update, PMLS-RM is not guaranteed delivery any more. In other words, we cannot decrease F without limitation.

$N = 300$, $D = 24$, $S = 1$m/s	Message Cost	Hop count dilation
DC-MR	4.49	1.08
PMLS-RM ($F = 9$s)	5.7	1.05
PMLS-RM ($F = 20$s)	5.13	1.06

Table 4.5. PMLS-RM: Increase F from 9s to 20s to reduce message cost

$N = 300$, $D = 14$, $S = 10$m/s	Success rate	Message cost	Hop count dilation
DC-MR	99.6%	22.03	1.17
PMLS-RM ($F = 9$s)	96.4%	11.01	1.37
PMLS-RM ($F = 5$s)	100%	13.73	1.26

Table 4.6. PMLS-RM: Decrease F from 9s to 5s to guarantee delivery

4.4 Drawbacks and Solutions

Every algorithm has its limitations and PMLS-RM is not an exception.

(1) No guaranteed delivery if all the neighbors disconnected to actor

As we stated before, PMLS-RM requires that the value of F is small enough so that at least one neighbor is connected to the actor between two successive location updates; otherwise, it cannot guarantee delivery. From the above simulation results, we find that, in general, PMLS-RM does not need a small F to keep good performance. But special scenarios do exist. One example is that when the actor moves fast, it has more chances to disconnect all of its neighbors before sending next location update message. That is why we need to decrease F from 9s to 5s to guarantee delivery when S = 10m/s. Another example is that in low-degree networks with big N ($N \geq 400$ when $D = 8$), due to the small transmission radius, the actor has more chances to disconnect all of its neighbors too even its moving speed is small. That is why we need to decrease F from 9s to 6s to guarantee delivery when S = 1m/s.

(2) Long routing path if out-of-date location information

If its knowledge about the position of the actor is much out of date, the source node has to follow a long snake-like path to the actor. Such things do happen especially when the source node is far away from the moving trajectory of the actor.

Possible solution: another recovery location update.

Every time the actor receives a data packet from a node, say u, it compares the hop count of this data packet to a threshold. If above the threshold, the actor sends an extra recovery location update message to u using GFG algorithm, including its current position. All the neighbors of intermediate sensors get the updated position of the actor by monitoring their retransmissions. When u receives this message, it sends one small message, notifying current position of the actor to all its neighbors as well, and none of its neighbors will forward it. This solution is implemented and the corresponding messages are counted in the simulation.

Chapter 5 Position-Maintained Location Service for Controllable Movement

In this chapter, we elaborate the other variant of PMLS, Position-Maintained Location Service for Controllable Movement (PMLS-CM).

5.1 An Example First

The basic idea of PMLS-CM is that after initial exchange of HELLO message, based on its current neighbor list, the actor first calculates the minimal broken time and the corresponding broken node, and then sends a location update message to the broken node before it loses connection to the latter. Similar with PMLS-RM, each sensor retransmits the location update message if using GFG algorithm, its forwarding nodes for the old and new *endpoints* (destination of current movement) are different; otherwise, it keeps silent. Moreover, the actor can change the endpoint halfway if receiving higher priority event. If endpoint changes, before the direction is modified, the actor uses GFG algorithm to send a recovery location update message to the old endpoint, indicating the position of new endpoint. So the data packets routed to the old endpoint can be redirected to the new endpoint.

Let us start with an example. In Figure 5.1, suppose the initial location of the actor is A_0 and the actor firstly wants to move to endpoint A_1 with certain selected speed. At A_0, the actor calculates the minimal broken time and finds that node u is the first broken node. When the actor moves to location B, before the link to node u is broken, the actor sends a location update message, including its current endpoint A_1, to all its neighbors.

All the sensors receiving the location update message will retransmit it if the forwarding nodes for the old and new endpoints are different; otherwise, they keep silent. And the sensors receiving the retransmitted message obey the same rule.

At location *A'*, the actor changes its direction and moves towards the new endpoint A_2. Before the change, using GFG algorithm, it sends a recovery location update message, including the position of the new endpoint A_2, to the old endpoint A_1. Since A_1 is only a location, not a real node, the recovery location update message will stop at the node geographically closest to A_1, which is node *b* in this example [RKYYEGS]. According to GFG, only after *b* uses face routing to forward the recovery location update message on the face enclosing A_1 (the face including nodes *b*, *c*, *d*, *e*, *f* and *g* in Figure 5.1) and receives the same message twice, *b* knows itself is the closest node to A_1 and stops routing. During the process of face transversal, all the nodes on the face know the position of the new endpoint A_2. The actor may change its endpoint several times.

When node *u* wants to send a data packet to the actor, it estimates the time needed for the actor to arrive at the endpoint A_1 and sets up a timeout. When the timeout expires, *u* routes the packet to A_1 using GFG algorithm. Due to the characteristic of GFG, the packet will reach one of the nodes, node *c* in Figure 5.1, on the face perimeter enclosing A_1 and is redirected to the new endpoint A_2 afterwards.

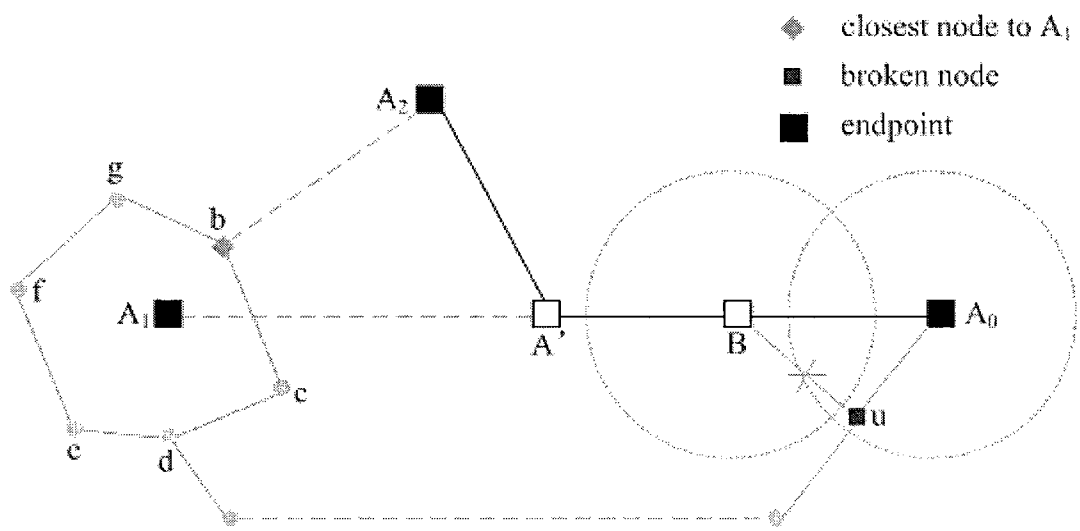

Fig. 5.1. PMLS-CM: an example

5.2 Description of Algorithm

The corresponding algorithm, PMLS-CM, is described as follows:

1. The actor and its neighboring sensors periodically exchange HELLO messages.
2. After initial HELLO message exchange, the actor calculates the *minimal broken time* and the corresponding broken node based on its neighbor list.
3. When the minimal broken time expires, the actor sends a LOCATION UPDATE message, including its current position, location of current *endpoint*, moving speed and current time, to all neighbors, indicating the broken node as well. In the meanwhile, it calculates the next minimal broken time and corresponding broken node.

   If the periodical HELLO message exchange starts before the expiry of minimal broken time, the actor stops current minimal broken timeout event and goes to step 2.
4. When a sensor receives a LOCATION UPDATE message, it records the positions of the actor and the endpoint, and moving speed first, then
   (a) If it is a new neighbor of the actor, and has not sent HELLO message to the actor before, it adds the actor to its neighbor list, and sends HELLO message to the actor. Afterwards, it retransmits the message.
   (b) If it is the designated broken node, it removes the actor from its neighbor list and retransmits the message.
   (c) If it is neither a new neighbor nor a designated broken node, but the forwarding nodes for the old and new endpoints are different, it retransmits the message; otherwise, it keeps silent.
5. When the actor receives the HELLO message from the new neighbor, it adds the new neighbor to the neighbor list, stops current minimal broken timeout event and goes to step 2.

6. If the actor receives a higher priority event and needs to change the endpoint, it sends a RECOVERY LOCATION UPDATE message regarding the current endpoint as the destination, including actor's current position, location of the new endpoint, new moving speed and current time.

   In the meanwhile, it stops current minimal broken timeout event and goes to step 2.

7. When a sensor receives the RECOVERY LOCATION UPDATE message, it records the positions of the actor and the new endpoint, and moving speed actor first, and then forwards the message using GFG algorithm.

   (a) If the sensor is the closest sensor among all the sensors to the old endpoint, it will receive the same message twice: one via greedy routing and the other via face routing. Afterwards, it knows it is the closest node and stops routing. During the face transversal, all the nodes on the face perimeter enclosing the old endpoint get the location of the new endpoint.

   (b) For each other node receiving the RECOVERY LOCATION UPDATE message, it forwards the message and all its neighbors get the updated information of the actor and the endpoint by monitoring its retransmission.

8. When a sensor, say u, wants to route a data packet to the actor, it first, based on the information of actor's current position, location of the endpoint, moving speed and time elapsed, calculates the time needed for the actor to arrive at the endpoint and sets its *waiting time* accordingly.

   When the waiting time expires, u routes the packet to the endpoint. The packet may be redirected to a new endpoint either upon arriving at one of the nodes on the face enclosing the old endpoint, or through a node which has the more up-to-date location of the endpoint. This kind of redirection may happen several times if the actor changes the endpoint more than once.

5.3 Pseudo Code

Again, we list the algorithm for the actor part and the sensor part separately.

Algorithm 2-a: PMLS-CM (actor part), actor *A*

```
1:  while true do
2:      Calculate the minimal broken time and corresponding broken node based on current neighbor list
3:      if minimal broken time expires then
4:          A sends LOCATION UPDATE message to all neighbors including its current position,
            location of endpoint, moving speed, current time and designated broken node and goes to 2
5:      else if receiving HELLO message from new neighbor or periodical HELLO message exchange then
6:              A stops current minimal broken timeout event and goes to 2
7:      else if receiving a higher priority event then
8:              A sends RECOVERY LOCATION UPDATE to old endpoint including current position,
                location of new endpoint, new moving speed and current time;
                A stops current minimal broken timeout event and go to 2.
```

Algorithm 2-b: PMLS-CM (sensor part), sensor *u*

```
1:  while true do
2:      check the type of message m received
3:      switch type of message do
4:          case type = 'LOCATION UPDATE'
5:              if u is new neighbor AND has not sent HELLO message to actor before then
6:                  u sends HELLO message to A;
                    u adds A to its neighbor list and retransmits
7:              else if u is designated broken node then
8:                      u moves A from neighbor list and retransmits
9:              else if the forwarding nodes of u for old and new endpoints are different then u retransmits
10:         case type = 'RECOVERY LOCATION UPDATE'
11:             if receiving same message twice then
12:                 u knows itself is the closest node to old endpoint and stops routing
```

13: **else**

14: u forwards message according to GFG algorithm and all its neighbors get the updated position of actor by monitoring u's retransmission

5.4 Proof of Guaranteed Delivery

Theorem 2. *Suppose actor always keeps connected to the sensor network. PMLS-CM algorithm provides guaranteed delivery to routing from sensor to the slowly mobile actor for any connected sensor network modeled as unit disk graph.*

Proof. Suppose actor A has changed directions several times, to endpoints $A_1, A_2, \ldots$ respectively. Its initial position is A_0 and current endpoint is A_k. Obviously, every sensor gets one position of endpoint, initially A_1 or one of later positions of the endpoint.

Suppose sensor u with endpoint knowledge A_1 sends a packet to A. Based on Lemma 1, the recovery location update message initiated by the actor to A_1 before changing to endpoint A_2, is routed to the closest node to A_1, say w_1, among all sensors. In other words, w_1 has endpoint knowledge A_2 now. Based on the same Lemma 1, the data packet from u will also be routed to w_1 and redirected to A_2 afterwards. Thus the advance toward A_k is made, and iteratively further steps can also be made. That is, if $i < k$, w_i will acquire the endpoint knowledge A_{i+1} through recovery location update message sent by the actor before moving toward endpoint A_{i+1}. So the packet arriving at w_i will be redirected to A_{i+1}. Eventually, the packet will be forwarded to a node w_k that is closest to the current endpoint A_k and forwards the packet to the actor, as illustrated in Figure 5.2. Since all the nodes on the face enclosing endpoint A_i also get the updated location of the new end A_{i+1} during the forwarding of recovery location update message, the data packet of u can be redirected to A_{i+1} through any of these nodes, including w_i. □

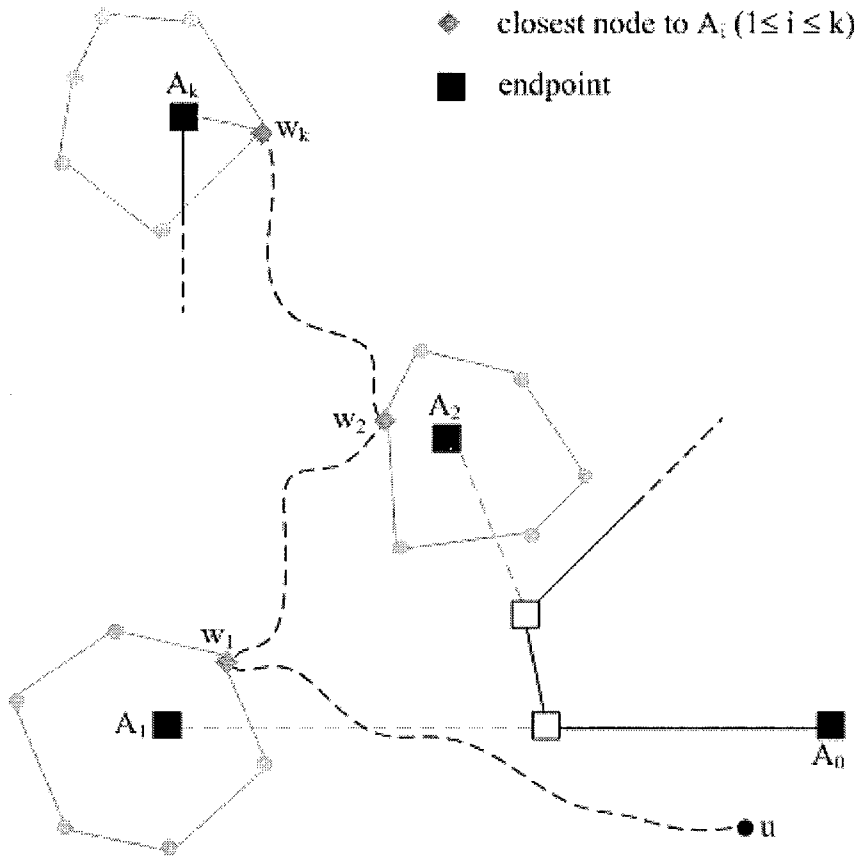

Fig. 5.2. PMLS-CM: guaranteed delivery in connected sensor network modeled as UDG graph.

Chapter 6 Performance Evaluation of PMLS-CM

We use the same simulation environment and parameters as Chapter 4 to compare the performance between PMLS-CM and DC-MR. We do not need parameter F (period of HELLO message exchange), but we need to add one new parameter, E, indicating the ratio of sudden endpoint change to normal endpoint change. Here, *sudden endpoint change* means that the actor changes the endpoint before arriving at it, while *normal endpoint change* means that actor changes the endpoint after arriving at it.

In the following experiments, unless clearly stated, we set maximum moving speed S = 1m/s, E = 1:2 and $L = a$ for $N = a \times 100$.

6.1 Message Cost

In this section, we compare the message cost between PMLS-CM and DC-MR in terms of average node degree D and node number N, respectively.

Figure 6.1 shows that, when N = 300, the message cost of DC-MR decreases while D increases, but the message cost of PMLS-CM increases while D increases. For small D, such as D = 8, the message cost of PMLS-CM is only 50% of that of DC-MR. The reason is that in low-degree networks, in DC-MR, almost every node in the network needs to forward the location update message once, so the message cost of DC-MR in low-degree networks is high.

With the increase of D, PMLS-CM's lead over DC-MR gets smaller. Finally, DC-MR defeats PMLS-CM in dense networks, same as PMLS-RM is defeated by DC-MR in dense networks. The reason is actually same too. When D increases, the set of Multipoint Relay gets smaller and thus number of nodes needed to forward the location update message in DC-MR decreases. On the contrary, for same value of L, more nodes are involved and needed to forward location update messages, so the message cost of PMLS-CM increases.

Different from PMLS-RM, the message cost curve of PMLS-CM is not as smooth as the curve of PMLS-RM, but has some vibration. The reason is that the performance of PMLS-CM is more vulnerable to the position of the source node. We can find such kind of phenomenon in almost all the following graphs. We will detail this problem later.

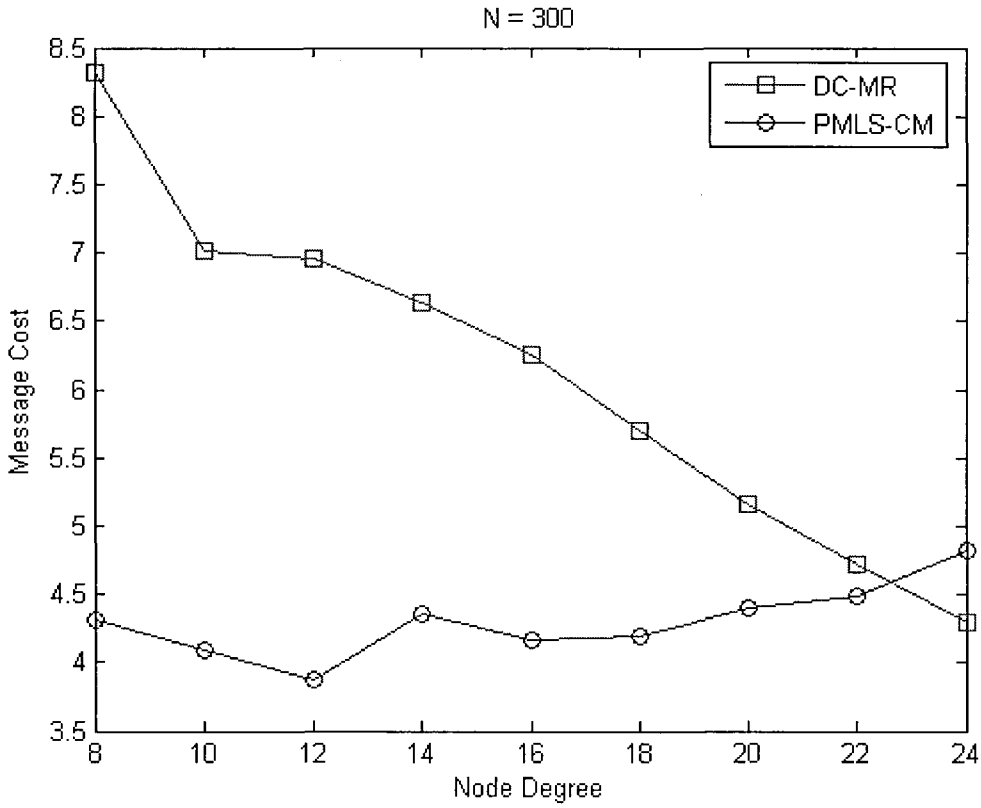

Fig. 6.1. PMLS-CM vs. DC-MR: message cost at various node degrees.

Figure 6.2 shows that, when D = 14, the message cost of both PMLS-CM and DC-MR increases while N increases. In all the cases, PMLS-CM is superior to DC-MR, 61% to 34% reduction for N = 100 to 500. The reason why the message cost of PMLS-CM is notably better than that of DC-MR when N = 100, is that in low-number networks, almost all the 1-hop neighbors is in the set of Multipoint Relay, which leads to the high message cost of DC-MR. When N gets bigger, the set of Multipoint Relay gets smaller, which offsets the impact of the increasing N, so the difference between DC-MR and PMLS-CM is smaller. When N is big

enough, for both PMLS-CM and DC-MR, the change of message cost according to node number is very small, so the difference between them remains stable.

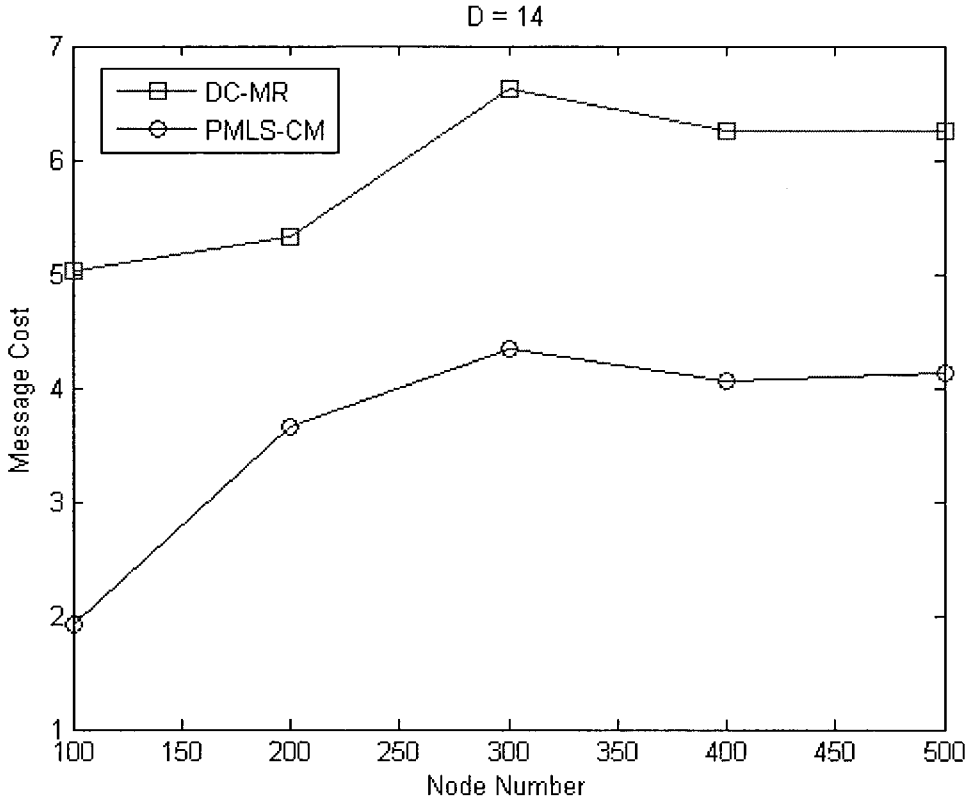

Fig. 6.2. PMLS-CM vs. DC-MR: message cost at various node numbers.

In summary, except in high dense networks, PMLS-CM shows apparently better performance than DC-MR in message cost. PMLS-CM's lead over DC-MR is large in either low-degree networks or sparse networks. We will elaborate these two scenarios later.

6.2 Hop Count Dilation

In this section, we compare the hop count dilation between PMLS-CM and DC-MR in terms of average node degree *D* and node number *N*, respectively.

Figure 6.3 shows that, when $N = 300$, the hop count dilation of both DC-MR and PMLS-CM decreases while D increases, because the performance of GFG algorithm gets better with the increase of D and very close to SP algorithm in dense networks [DSW]. In all the cases, DC-MR is superior to PMLS-CM but the difference between them gets smaller with the increase of D and remains very close in dense networks. The reason PMLS-CM always has longer routing path than DC-MR is that in PMLS-CM, the source nodes need to wait for the actor's arrival at expected endpoint. If the actor changes the endpoint halfway, they will miss the actor and thus the length of routing path is increased. If the actor changes the endpoint more frequently or moves more quickly, source nodes have more chances to miss the actor at expected endpoint, which results in even longer routing path. We will see such phenomena in the following analysis.

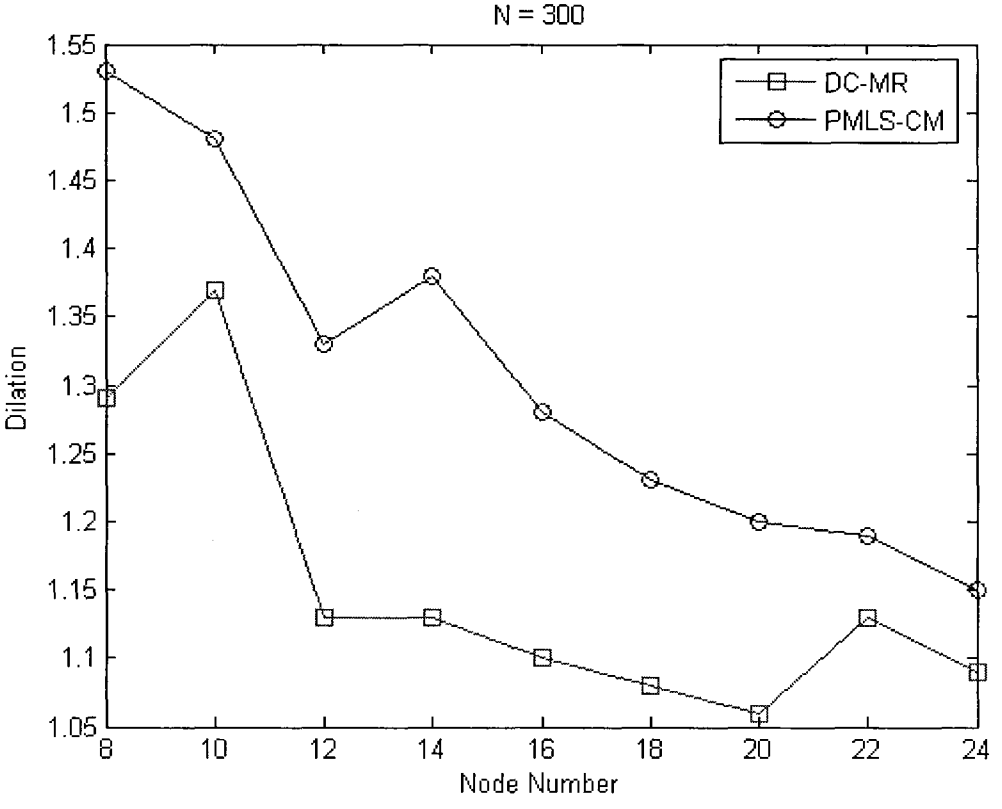

Fig. 6.3. PMLS-CM vs. DC-MR: dilation at various node degrees.

Figure 6.4 shows that, when D = 14, the hop count dilation of both DC-MR and PMLS-CM decreases when N increases. Since the size of the set of Multipoint Relay depends more on D than on N, the performance of DC-MR only slightly changes when N increases. In all the cases, DC-MR is superior to PMLS-CM but the difference between them is slightly getting smaller when N increases. The reason is that with the increase of N, the source nodes have more opportunities to get the information of current endpoint thus shorten the average length of routing path.

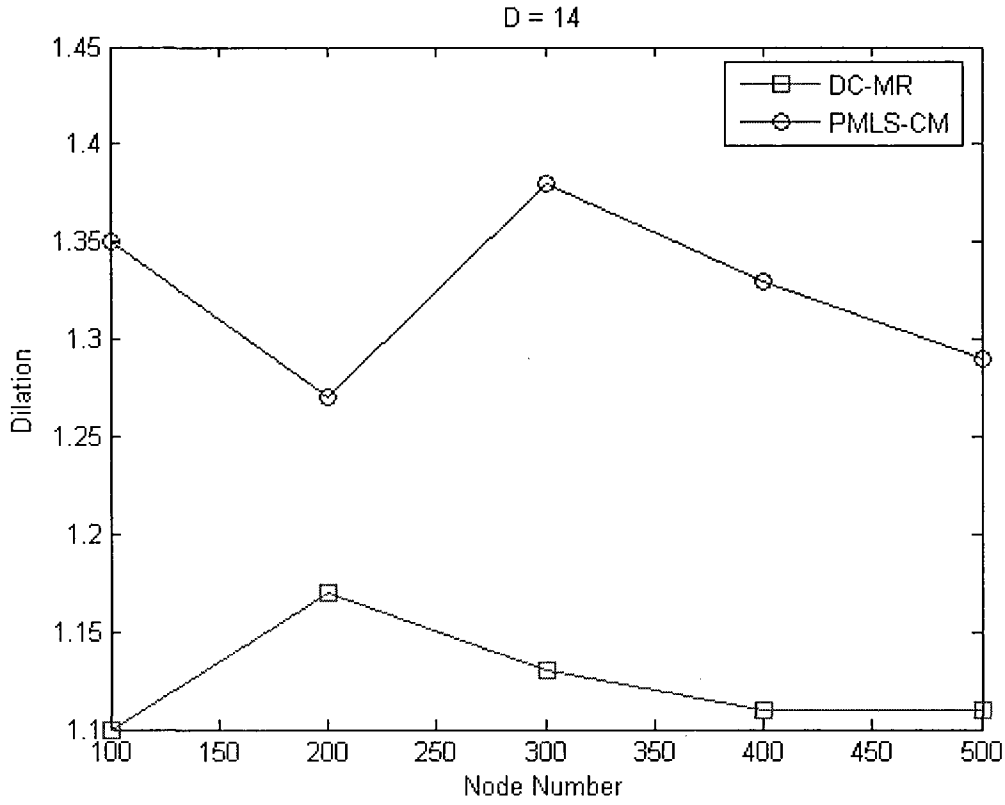

Fig. 6.4. PMLS-CM vs. DC-MR: dilation at various node numbers.

In summary, DC-MR is superior to PMLS-CM in hop count dilation but the difference is small. Moreover, the gap is either getting smaller or remains very close with the increase of node number or node degree.

6.3 Impact of Moving Speed

In this section, we study the impact of moving speed on both message cost and hop count dilation. We compare four values of maximum speed: 1m/s, 2m/s, 5m/s and 10m/s. To focus more on the impact of moving speed and reduce the affect of node number and node average, we fix $N = 300$ and $D = 14$.

Figure 6.5 shows that, when $N = 300$ and $D = 14$, the message cost of both DC-MR and PMLS-CM increases when S increases. The reason is that with the increase of S, for DC-MR, actor has more opportunities to move outside certain circle, thus leading to more frequent location updates; for PMLS-CM, more edges are broken during the same period of time, thus leading to higher message cost as well. In all the cases, PMLS-CM is superior to DC-MR. The difference between them gets smaller with the increase of D and remains stable when D is big enough.

Figure 6.6 shows that, when $N = 300$ and $D = 14$, the hop count dilation of PMLS-CM first decreases then increases when S increases. The reason is that when S increases but not big enough, there are more location updates during the same amount of time, so source nodes have more opportunity to get the position of current endpoint. When S is big enough, if the actor changes the endpoint halfway, it is easier for source nodes to miss the actor at previous endpoint and lengthen the length of routing path.

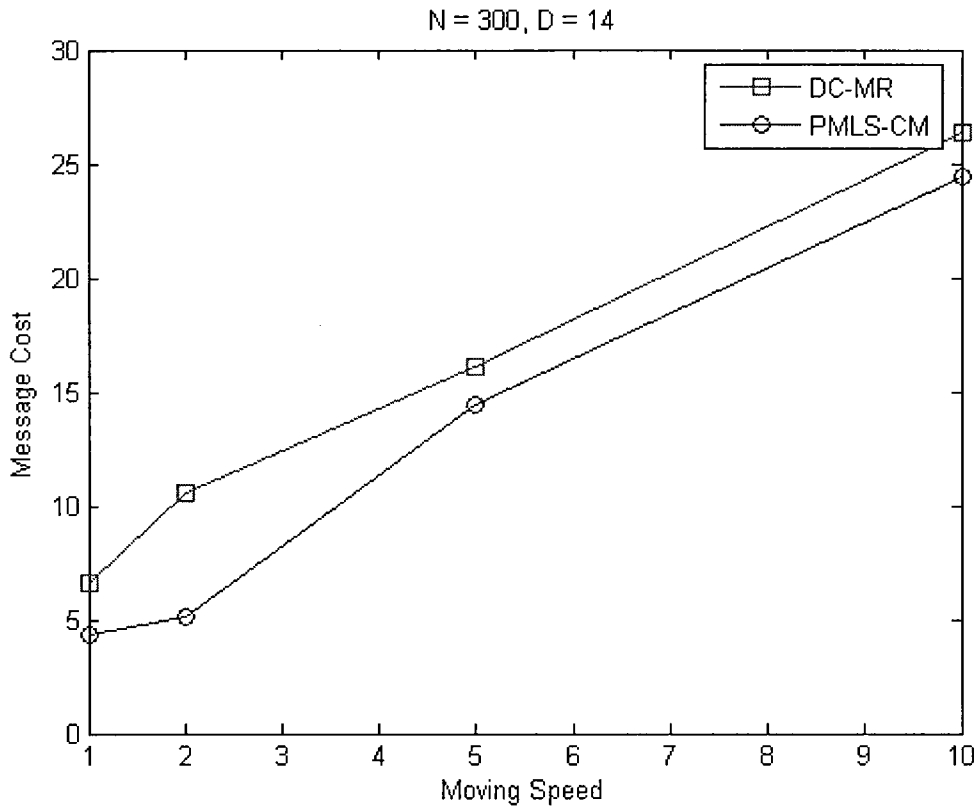

Fig. 6.5. PMLS-CM vs. DC-MR: message cost at various moving speeds

On the contrary, the hop count dilation of DC-MR first decreases when S is small then increases when S gets bigger and keeps stable when S is big enough. The reason is that when S = 2m/s, compared with the scenario when S = 1m/s, there are more location updates and it is easier for source node to get the location of the endpoint, resulting in the reduction of hop count dilation. When S gets big, actor travels more distance during the same period of time, for the same source node, it has to travel longer distance to successfully send data packet to the actor, thus leading to bigger hop count dilation. In all the cases, DC-MR is superior to PMLS-CM but the difference is small.

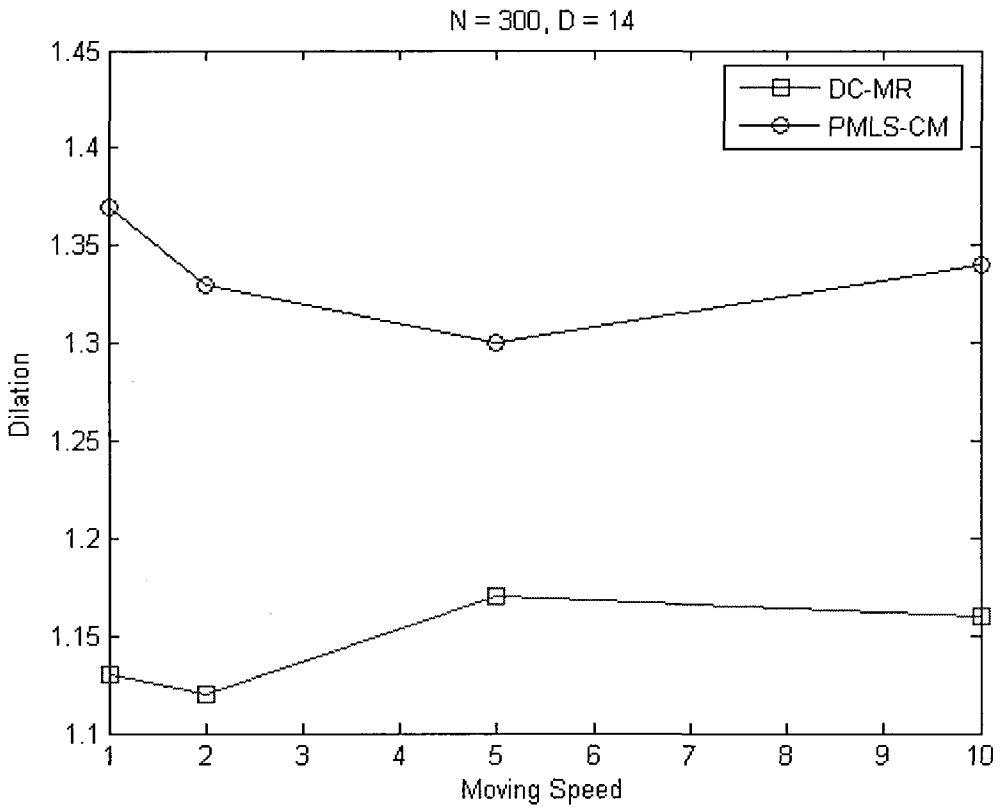

Fig. 6.6. PMLS-CM vs. DC-MR: dilation at various moving speeds

Figure 6.7 shows that, when N = 300 and D = 14, both PMLS-CM and DC-MR cannot reach 100% success rate when S is big. The reason for no guaranteed delivery in PMLS-CM is that source node may always follow up the moving trajectory of the actor but can only catch up with it at the final endpoint if the actor changes the endpoint several times. Such kind of routing task will take a long time and cannot finish before the endpoint of simulation time. In other words, it 'fails'. This is actually a drawback of PMLS-CM algorithm. We will specify it in later section.

And the reason for the routing fail of DC-MR at high speed is the same for PMLS-CM. We may find that in Figure 6.7, compared with the result when S = 5m/s, the success rate of DC-MR when S = 10m/s is better. The reason is that if the actor moves around certain area, even source node miss the actor at expected endpoint, its data packet may probably still be forwarded

to the actor nearby in short time. In other words, mobility model plays a more important role when moving speed is big.

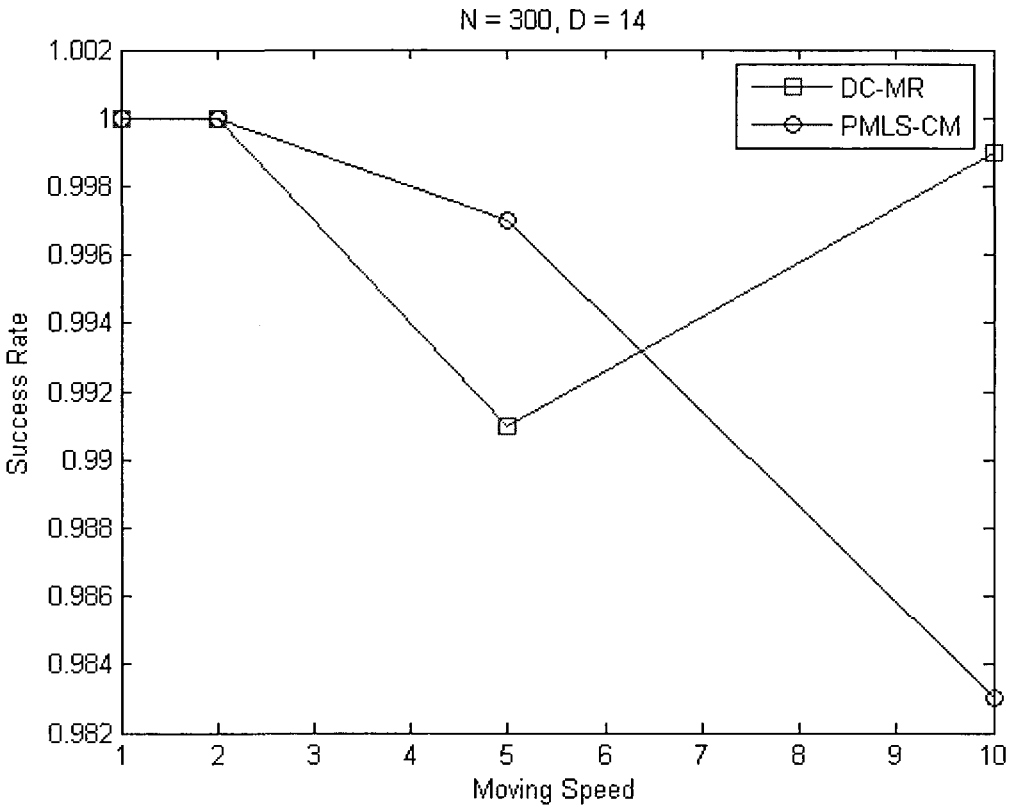

Fig. 6.7. PMLS-CM vs. DC-MR: success rate at various moving speeds

In summary, when moving speed of the actor increases, PMLS-CM has better performance than DC-MR in message cost, but DC-MR has better hop count dilation. Both PMLS-CM and DC-MR cannot guarantee delivery when the speed is high.

6.4 Performance in Low-degree Networks

In this section, we study the performance of DC-MR and PMLS-CM in low-degree networks. We set $D = 8$ and vary N from 100 to 400.

Figure 6.8 shows that, when D = 8, the message cost of both DC-MR and PMLS-CM increases while the N increases. In all the cases, PMLS-CM is notably better than DC-MR, over 40% reduction. The reason is that in such scenario, the set of Multipoint Relay in DC-MR is almost the same as the 1-hop neighbor set, especially in the sparse networks, so the performance of DC-MR has high message complexity. In contrast, since the level of message forwarding is small and fixed in PMLS-CM, only small portion of nodes will retransmit the location update messages, so the message cost of PMLS-CM is small.

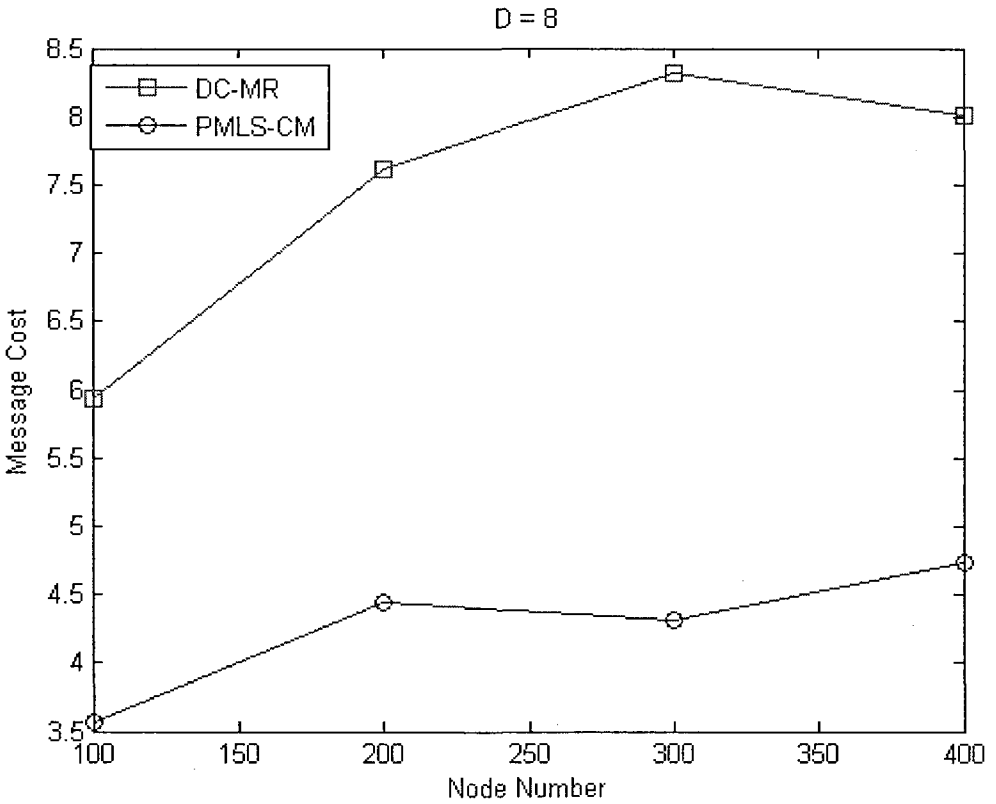

Fig. 6.8. PMLS-CM vs. DC-MR: message cost in low-degree networks

Figure 6.9 shows that, when D = 8, in all the cases, the hop count dilation of DC-MR is superior to that of PMLS-CM. The reason is that in low-degree networks, compared with DC-MR, the number of the nodes having the location of current endpoint in PMLS-CM is smaller. When the actor changes endpoint, it is easier for the source node to miss the actor at previous endpoint

and results in a longer routing path. On the contrary, DC-MR uses the flooding-like mechanism, so the source node has more opportunities to get the location of current endpoint, though at the price of high message cost.

We also find that the hop count of both DC-MR and PMLS-CM vibrates when N increases. The reason is that in low-degree networks, the hop count dilation is more vulnerable to the position of source node, as we pointed out before.

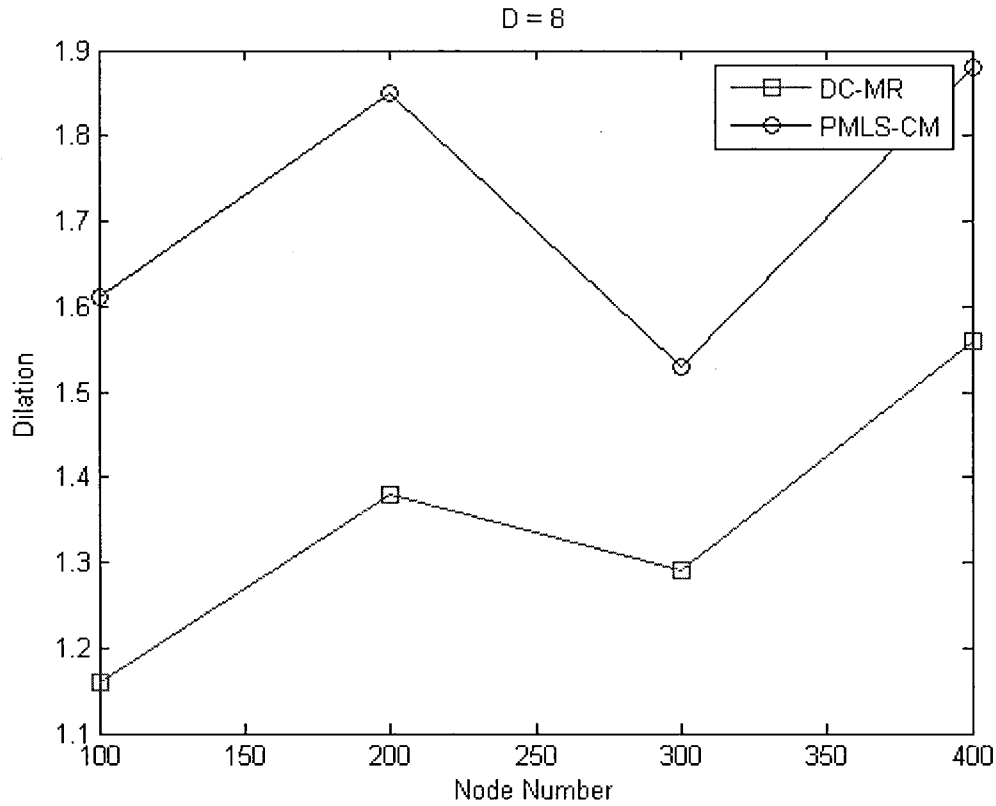

Fig. 6.9. PMLS-CM vs. DC-MR: dilation in low-degree networks

Figure 6.10 shows that, when D = 8, the success rate of both DC-MR and PMLS-CM slightly decreases when node number increases. When $N \leq 300$, PMLS-CM guarantees delivery. However, as the increase of N, if the actor changes the endpoint several times, the source node has more chances to follow up the moving trajectory of the actor and catch up with it at the final

endpoint. Thus the routing time becomes pretty long in such cases and beyond the scope of simulation duration, leading to the failure of routing task finally.

The reason why DC-MR for controllable movement cannot guarantee delivery in low-degree networks is the same as the reason why DC-MR for random movement cannot, as indicated in Figure 4.9.

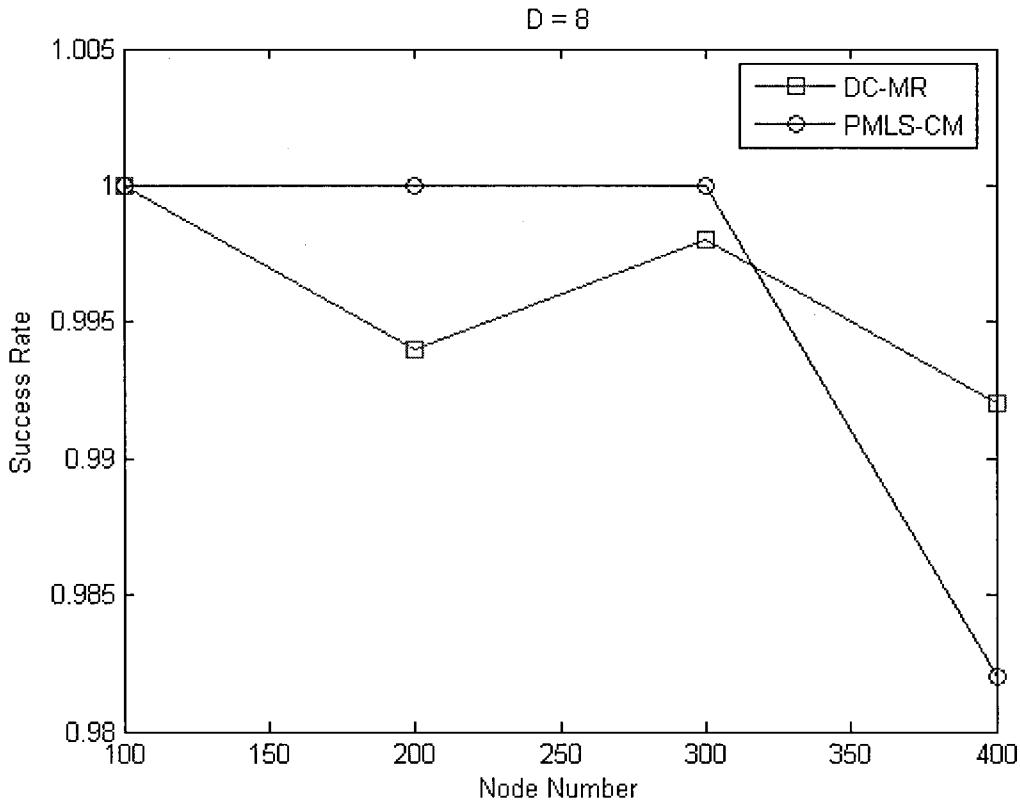

Fig. 6.10. PMLS-CM vs. DC-MR: success rate in low-degree networks

In summary, in low-degree networks, compared with DC-MR, the performance of PMLS-CM is much better in message cost, slightly better in success rate when N is not big but inferior to DC-MR in hop count dilation.

6.5 Performance in Dense Networks

In this section, we study the performance of DC-MR and PMLS-CM in dense networks.

As illustrated in Figure 6.1, DC-MR performs better than PMLS-CM in dense network. Here, we only use the case of $N = 300$ and $D = 24$ as the example and give one possible solution.

Same as the second method used for PMLS-RM, we can decrease the level of message forwarding (L), from 3 to 2, to reduce the message cost of PMLS-CM. Table 6.1 shows the same improvement as we expect: the decrease of L apparently reduces the message cost of PMLS-RM with no change of guaranteed delivery and slight increase of hop count dilation.

$N = 300, D = 24, S = 1$m/s	Message Cost	Hop count dilation
DC-MR	4.3	1.09
PMLS-CM ($L = 3$)	4.82	1.15
PMLS-CM ($L = 2$)	3.67	1.22

Table 6.1. PMLS-CM: decrease L to reduce message cost in dense networks

In summary, in dense networks, the performance of DC-MR is generally better than that of PMLS-CM. However, we can still manage to make PMLS-CM superior to DC-MR by decreasing the level of message forwarding properly.

6.6 Impact of Ratio of Sudden Endpoint Change to Normal Endpoint Change

In this section, we study the impact of ratio of sudden endpoint change to normal endpoint change (E) on all the compared metrics.

Figure 6.11 shows that, when $N = 300$ and $D = 14$, the message cost of PMLS-CM increases when E increases, while the message cost of DC-MR slightly increases first and then decreases. The reason for the decrease of the message cost in DC-MR is that when E increases, the actor needs to change its endpoint and moving direction more often. In other words, the actor

moves a short distance in each direction. So DC-MR only needs to send location updates within small circles, thus generates less message cost. On the contrary, PMLS-CM needs to send more RECOVERY LOCATION UPDATE messages, resulting in more message cost. However, when $E \leq 1{:}1$, PMLS-CM is still superior to DC-MR.

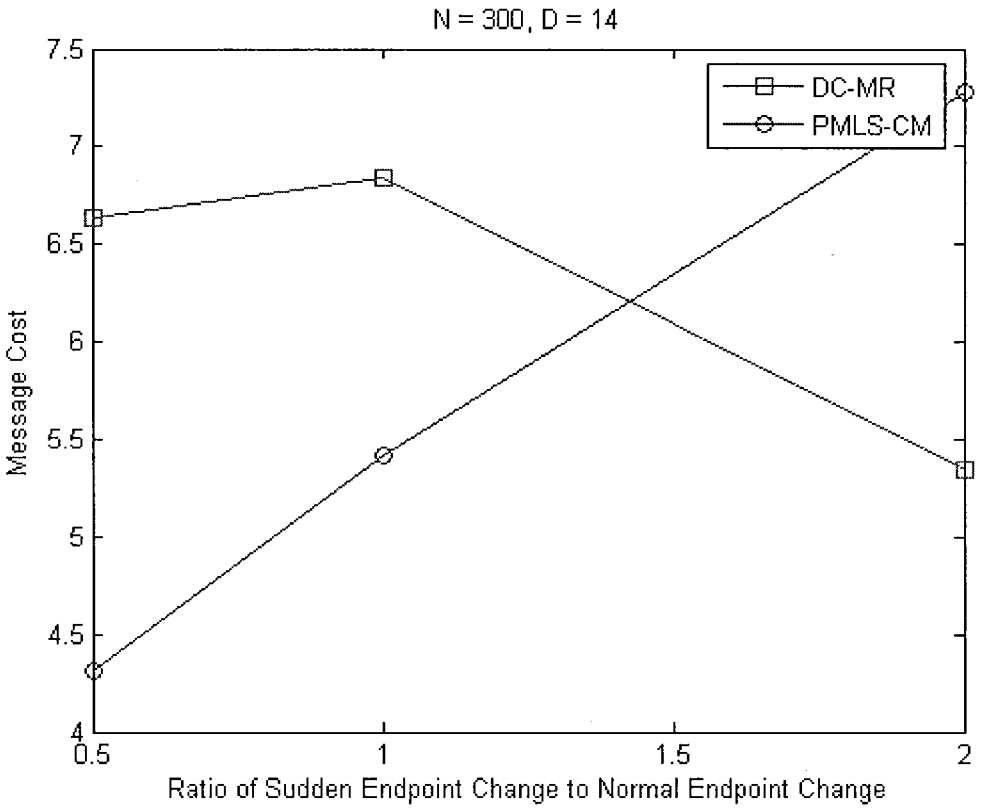

Fig. 6.11. PMLS-CM vs. DC-MR: message cost when increase E from 1:2 to 2:1

In Figure 6.12, when $N = 300$ and $D = 14$, the hop count dilation of both DC-MR and PMLS-CM shows the same trend as the message cost in Figure 6.11. The reason the hop count dilation of PMLS-CM slightly decreases when $E > 1{:}1$, is that PMLS-CM is not guaranteed delivery in this scenario, so only successful routing tasks, which are less than 100, are counted. In all the cases, DC-MR is superior to PMLS-CM but the difference between them does not change much.

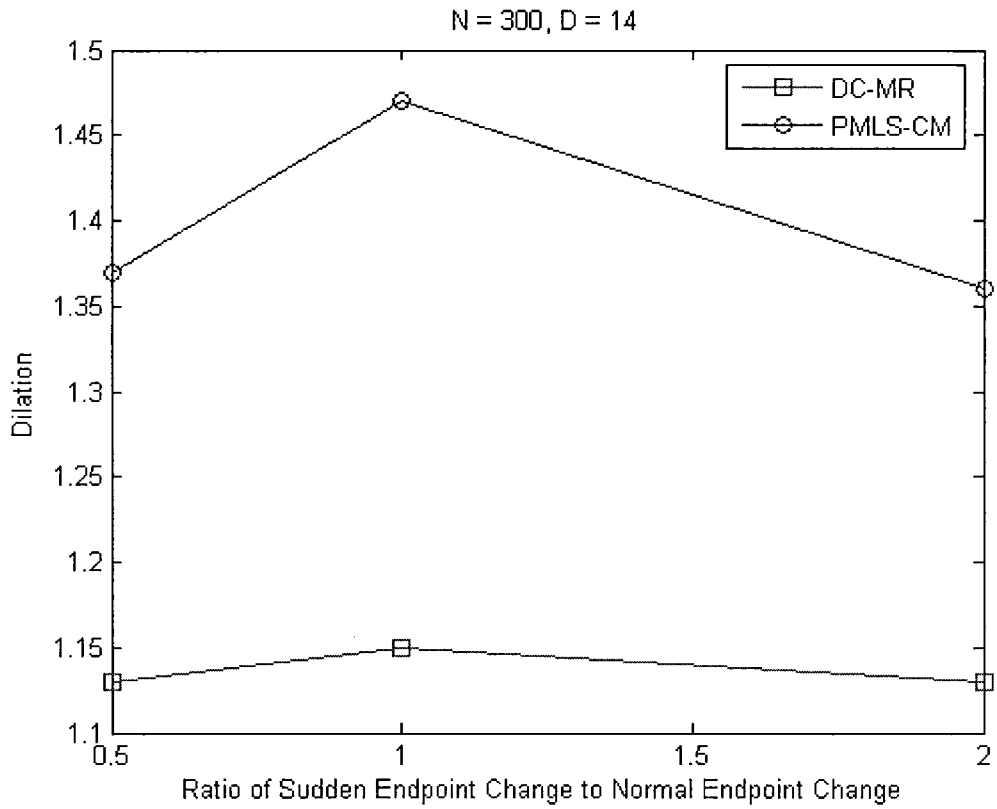

Fig. 6.12. PMLS-CM vs. DC-MR: dilation when increase E from 1:2 to 2:1

Figure 6.13 shows the change of success rate when E increases. DC-MR guarantees delivery because when E increases, due to more frequent change of the endpoint, the actor only moves a short distance in each direction, which makes the guaranteed delivery of DC-MR easier. On the contrary, PMLS-CM cannot guarantee delivery when E is big because the source node is more likely to miss the actor at its expected endpoint and need a long time and a long path to route a data packet to it, which reduces the average success rate.

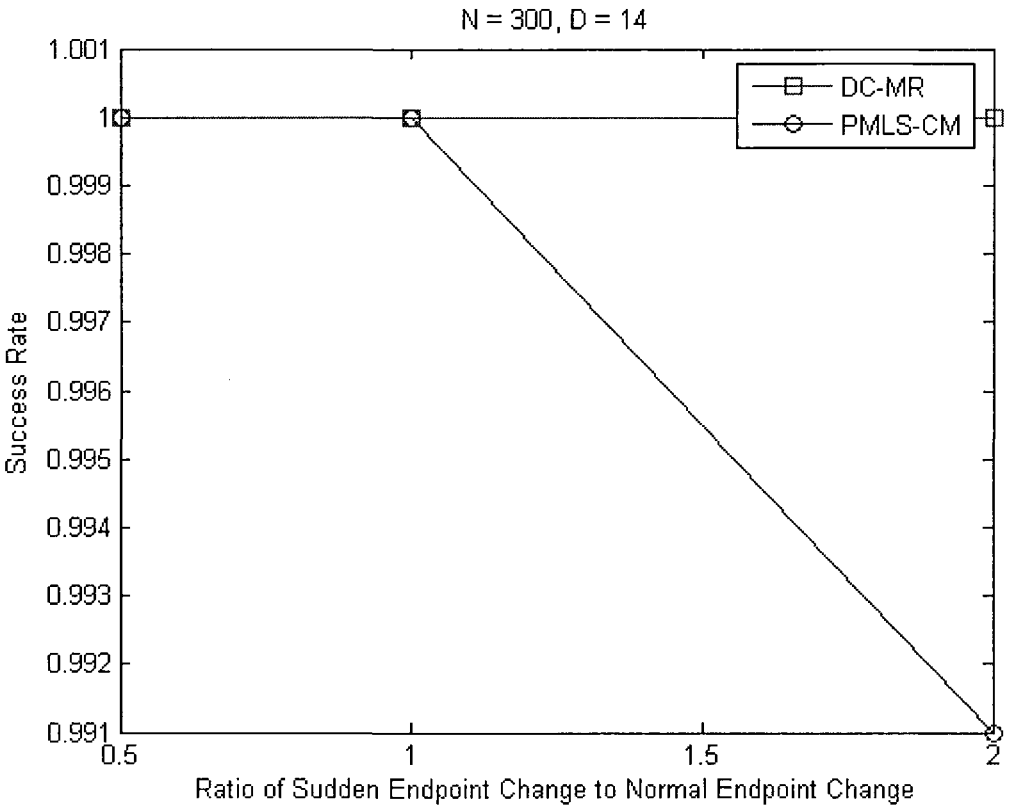

Fig. 6.13. PMLS-CM vs. DC-MR: success rate when increase E from 1:2 to 2:1

In summary, when E is no bigger than 1:1, PMLS-CM has better performance than DC-MR in message cost. However, when the ratio gets bigger, PMLS-CM is inferior to DC-MR in all the three metrics.

6.7 Other Impact Factors

The following factors also work for PMLS-RM and DC-MR for random movement. Furthermore, the final result of simulation is a combined function of all the factors, not only one of them or several of them.

6.7.1 Impact of Position of the Source Node

The position of the source node certainly has some impact on the hop count dilation and success rate of PMLS-CM. If the position of the source node is far away from the current endpoint and makes it difficult for source node to get the updated information, except in dense networks, the source node usually has to follow up the moving trajectory of the actor and takes a long time to catch up with the actor at the final endpoint. The routing path will be the snake-like dashed curves in Figure 5.2. If the routing time is beyond simulation duration, PMLS-CM is not guaranteed delivery.

The position of the source node has same, but generally smaller impact on the performance of DC-MR. The reason is that DC-MR uses flooding-like mechanism so there are more nodes getting the updated location of the endpoint, thus the source node has more opportunities to catch up with the actor before it moves to next endpoint. But in low-degree networks with big node number or if actor moves toward one direction for a long time, the situation becomes worse.

Inspiringly, the impact of the position of the source node gets smaller with the increase of node degree. In low-degree networks, due to low network density, its impact will cause vibration of the performance, as indicated in Figure 6.9. In dense networks, the source node has much more chances to get the location of current endpoint, so its impact becomes very small. That is why the hop count dilation of PMLS-CM is very close to that of DC-MR in such scenario, as illustrated in Figure 6.3.

6.7.2 Impact of UDG Graph

The shape of UDG graph, more precisely, the distribution of nodes in the UDG graph, has some impact on the message cost of PMLS-CM. For example, suppose nodes are located densely in certain area. Since the actor sends one location update message for every broken edge, if the actor moves around this area frequently, PMLS-CM will have higher message cost than the average. If the actor seldom moves around this area, the message cost will much less.

Since the UDG graphs used in our simulation are randomly generated and nodes in it are generally evenly distributed, we do not see obvious impact of UDG graph on the simulation results. If the underlying UDG graphs are mostly unevenly distributed in certain application, we may increase the number of broken edges for triggering location updates, for example, from 1 to 2, to reduce the message cost.

UDG graph also has the same, but smaller impact on the performance of DC-MR. The reason is that the trigger mechanism of location update in DC-MR is different. It is the moving distance, not number of broken edges, which determines the time for location updates. If the actor does not move outside certain circle, even many edges are broken during its movement, there is still no location update at all.

6.7.3 Impact of Mobility Model

Mobility model always has impact on the performance of all the algorithms, although different algorithms may have different preferences.

As we mentioned above, when the actor moves around certain area, but changes the endpoint frequently, the performance of DC-MR may be better than PMLS-CM in every compared metric. DC-MR has low message cost because the actor only needs to send location update messages to the circles with small radii. The hop count dilation will not be big either because the source node can easily route data packets to the actor in short time even if its information about current position of the actor is inaccurate. And the success rate is high, usually 100%, for the same reason. For PMLS-CM, the situation is totally different. The message cost will not be reduced because the actor sends one location update for every broken edge in PMLS-CM. Moreover, because the source node usually can only catch up with the actor at the final endpoint when the actor changes the endpoint often, the long routing path and routing time will cause high hop count dilation and not guaranteed delivery as well.

Naturally, DC-MR also has its own limitations. When the actor moves towards certain direction for a long time, contrarily, DC-MR may have worse performance than PMLS-CM in

every compared metric. DC-MR will have high message cost because in such scenario, the area needed to be updated gets bigger when the actor moves farther away, which will be the whole network eventually. If the source node is far away from the actor, it needs to travel a long distance to catch up with the actor, which leads to a big hop count dilation. If the routing time is too long and beyond simulation time, it can reduce the success rate too. In contrast, since the level of message forwarding is only related to node number, such kind of mobility model will not cause dramatic increase of message cost, thus has smaller impact on PMLS-CM.

6.8 Drawbacks and Solutions

PMLS-CM has its weaknesses as well.

(1) Long routing time and high buffer requirement if changing endpoint frequently

In PMLS-CM, in order to wait the actor for reaching the expected endpoint, every sensor needs to buffer the data packets for some time, which increases delay and requires higher buffer capacity. In the situation where the actor has to change the endpoint frequently or the application has tight delay-binding, this may not be acceptable.

Possible solution1: adding intermediate endpoints

To reduce the routing time and high buffer requirement, we can divide the distance between the start point of current movement (A) to the endpoint of current movement (B) into small segments according to some rules, for example, the length of each segment is equal to one tenth of the distance between A and B. At the beginning, A is the initial position of the actor and B is the location of current endpoint. If the actor does not change the endpoint, when the actor arrives at current endpoint, A is changed to the location of current endpoint and B is the location of next endpoint. If the actor decides to change endpoint before arriving at B, A is changed to the current position of the actor where direction change happens and B is the location of the new endpoint.

By this means, we create several *intermediate endpoints*. Then, instead of routing to the final endpoint, sensors can route to the nearest intermediate endpoint where the actor may be currently located, reducing both buffer size and delay. However, if the actor changes the endpoint halfway, we need to send RECOVERY LOCATION UPDATE message to all the intermediate endpoints which are closer to previous endpoint than the actor, so the message cost of PMLS-CM will increase.

Figure 6.14 gives an example. The initial position of the actor is A_0. There are two intermediate endpoints, M_1 and M_2, between A_0 and endpoint A_1. When the actor changes the endpoint to A_2 at position A', it only needs to send a RECOVERY LOCATION UPDATE message to M_2 and A_1 respectively. And the forwarding of these two messages will be stopped at the node u_2 and w_1 respectively. For broken node u, it sends data to M_2 instead of to A_1 now.

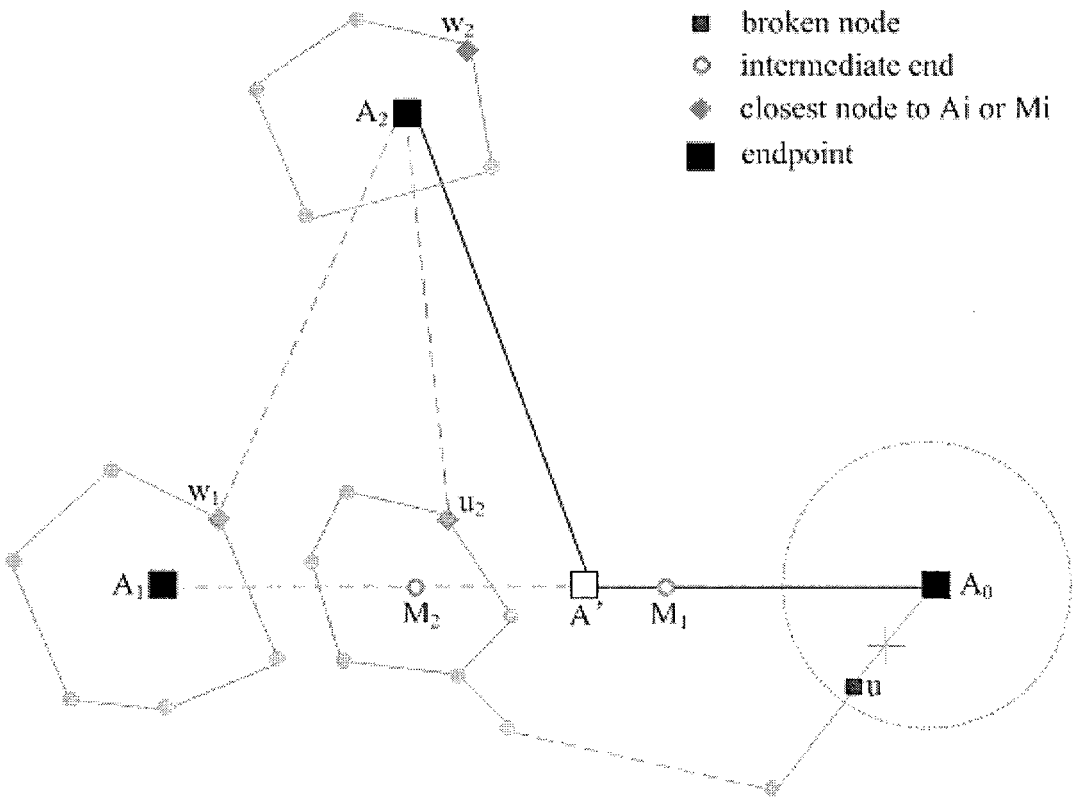

Fig. 6.14. PMLS-CM: add intermediate endpoints to improve performance

Note that if the distance between A and B is not long, the distance between two successive intermediate endpoints can be set to the communication radius (R) (the last segment can be shorter). This length ensures that the actor receives a message while moving, no matter the message is sent to its previous intermediate endpoint or following intermediate endpoint (due to wrong estimation of actor's location).

Possible solution2: routing to mobile actor directly

If the time for endpoint change has some principles, for example, between 1PM and 3PM, we may use this simpler method below.

If the actor is on its way to the endpoint and it is supposed not to change its direction halfway, the source node can calculate the current position of the actor based on the known information and send data packets directly to the actor. Since the transmission speed is much high compared with the moving speed of the actor, such packets can always be successfully received by the actor. If the source node cannot make sure the movement of the actor, it has to wait and send data packets to the expected endpoint.

We may combine solution1 and solution2 to get better performance.

(2) Long routing path if out-of-date location information

If its knowledge about the location of current endpoint is much out of date, same as in PMLS-RM, the source node has to follow a long snake-like path to the actor. Such things do happen especially when the actor changes the endpoint several times and the source node is far away from the moving trajectory of the actor.

Possible solution: another recovery location update.

We can use the same solution in PMLS-RM as well. Every time the actor receives a packet from a node u, it compares the hop count of this data packet to a threshold. If above the threshold, the actor sends an extra recovery location update message to u using GFG algorithm, including it current position and location of current endpoint. In addition to forwarding this recovery location update message, every intermediate sensor also sends an extra message, containing current positions of the actor and the endpoint, to all its neighbors, but none of its

neighbors will retransmit it. When u receives this message, it sends an extra message, notifying updated positions of the actor and the endpoint to all its neighbors too, but none of its neighbors will forward it. This solution is implemented and cost is counted in the simulation.

Chapter 7 Conclusions and Future Work

Existing location services can be divided into two types: *Flooding-based* and *rendezvous-based.* Flooding-based protocols usually only include location update step, not destination search step, but their frequent network-wide location updates cause high message cost and make them not scalable. Examples of flooding-based protocols are Doubling Circle scheme and Request Zone scheme. Rendezvous-based protocols follow the three-phase routing framework and usually involve steps of both location update and destination search. Some of them guarantee delivery, like Quorum-based scheme; some of them do not, like Home-agent Based scheme.

Compared with the existing protocols, the two location service algorithms, PMLS-RM and PMLS-CM (collectively called PMLS), have many advantages. On the one side, compared with Flooding-based protocols, PMLS also only has location update step but with less message cost. Since not all of, but only the nodes with different forwarding nodes for the old and new positions of the actor (in PMLS-RM) or for the old and new endpoints (in PMLS-CM), need to retransmit the location update messages, the message cost of PMLS is much lower than DC-MR. For PMLS-RM, there is over 70% reduction in low-degree or sparse networks, from 20% to 70% reduction in normal-density networks and over 20% reduction in dense networks; For PMLS-CM, there is over 40% reduction in low-degree or sparse networks, from 30% to 60% reduction in normal-density networks and around 15% reduction in dense networks.

On the other side, compared with Rendezvous-based protocols, PMLS guarantees delivery but do not need destination search step. Moreover, to guarantee delivery, PMLS does not need to always traverse the outer face of the network, which is required by Quorum-based Scheme and generates high message cost. So, PMLS is applicable for large-scale WSAN with reasonable message cost and guaranteed delivery.

Same as every other algorithm, both PMLS-RM and PMLS-CM have their own limitations. PMLS-RM requires at least on neighbor is connected to the actor between two successive location updates; otherwise, it cannot guarantee delivery. That means the value of period of HELLO message exchange has a bound. In some cases, for example, when the actor moves with high speed, we have to set the period small enough to guarantee delivery but increase the message cost obviously at the same time.

For PMLS-CM, since source nodes need to wait the actor for reaching the expected endpoint, they have to buffer the packets for some time, which increases both routing time and buffer capacity. If the actor changes the endpoint frequently, the routing time may be out of the scope of the simulation time and cut the success rate of PMLS-CM at last. Adding several intermediate endpoints on the line between the start point of current movement and the endpoint of current movement and making source nodes route data to its nearest intermediate endpoint instead of the endpoint is one solution. If the actor does not change the endpoint frequently, source nodes can also calculate the current position of the actor based on its present knowledge and route data directly to the mobile actor. These two methods can all reduce the routing time considerably and need to be implemented and compared in the future.

Although PMLS is proposed as a location service algorithm, it has other applications. It can be used as a routing algorithm to a mobile actor/sink. It can also be used as a destination search step under the three-phase routing framework.

In this thesis, we assume that the actor uses the same transmission radius (R) as the sensors. In reality, the actor usually uses bigger radius to send short control messages, such as location update messages, and use shorter radius to send/receive data packets. The use of the bigger transmission radius can significantly reduce the message cost of PMLS. For example, suppose the transmission radius of the actor is $2R$ and the level of message forwarding is a for $N = a \times 100$. When $N = 100$, we do not need to forward any location update message at all. For $N = 200$, we only need to forward location update messages once.

At the same time, the message cost of DC-MR will be reduced too. We need to conduct more simulations to find out which one can benefit more from this increasing transmission radius.

When the size of network gets bigger, to further reduce the message cost, we can setup a dominating set from the sensors. Let V denote the set of nodes in the network. A subset $V_D \subseteq V$ is said to be dominating if each node either belongs to V_D, or has at least one neighboring node that belongs to V_D. Based on 1-hop or 2-hop positional information, many algorithms are proposed to establish a Connected Dominating Set (CDS). Once CDS is constructed, only the nodes belong to CDS retransmit the location update messages. Although the non-dominant nodes do not retransmit the messages, they can get the updated position of the actor from the neighboring dominant nodes.

WSAN with multiple actuators is another direction of the future work. Single actor is usually not enough for a large-scale WSN, so WSAN with multiple actuators, where all the actuators form a high-speed network between themselves and connection to the sink, attracts more interests recently. The biggest difference between single actor and multiple actors is that to get good hop count, sensors need to send data to its nearest actor which is changeable when all the actors are moving. Our research in this thesis definitely provides grounds for the further investigation in this 'hot' direction.

References

[APL] Konstantions N. Amouris, Symeon Papavassiliou and Miao Li, "A position based multi-zone routing protocol for wide area mobile ad-hoc networks", Proc. 49th IEEE Vehicular Technology Conference, pp. 1365-1369, 1999.

[BCSW] S. Basagni, I. Chlamtac, V. R. Syrotiuk and B. A. Woodward, "A distance routing effect algorithm for mobility (DREAM)", Proc. MOBICOM, 1998, pp. 76-84.

[BMSU] Prosenjit Bose, Pat Morin, Ivan Stojmenovic and Jorge Urrutia, "Routing with guaranteed delivery in ad hoc wireless networks", ACM Wireless Networks, Vol. 7, No. 6, November 2001, pp. 609-616.

[DSW] Susanta Datta, Ivan Stojmenovic and Jie Wu, "Internal node and shortcut based routing with guaranteed delivery in wireless networks", Cluster Computing, Vol. 5, No. 2, April 2002, pp. 169-178.

[F] Gregory G. Finn, "Routing and Addressing Problems in Large Metropolitan-Scale Internetworks", Research Report ISU/RR-87-180, Inst. for Scientific Information, March 1987.

[FS] Hannes Frey and Ivan Stojmenovic, "On Delivery Guarantees of Face and Combined Greedy-Face Routing Algorithms in Ad Hoc and Sensor Networks", The Twelfth ACM Annual International Conference on Mobile Computing and Networking MOBICOM, Los Angeles, Sept. 23-29, 2006, pp. 390-401.

[GGHZZ] J. Gao, L. J. Guibas, J. E. Hershberger, L. Zhang, and A. Zhu, "Geometric spanner for routing in mobile networks," in 2nd Symposium on Mobile Ad Hoc Networking and Computing, Oct. 2001, pp. 45–55.

[GS] K. R. Gabriel and R. R. Sokal, "A new statistical approach to geographic variation analysis," Systematic Zoology, vol. 18 (3), pp. 259–278, 1969.

[JPS] Rahul Jain, Anuj Puri, and Raja Sengupta, "Geographical Routing Using Partial Information for Wireless Ad Hoc Networks", IEEE Pers. Commun., Feb. 2001, pp. 48-57.

[J-Sim] http://sites.google.com/site/jsimofficial/.

[JT] J. W. Jaromczyk and G. T. Toussaint, "Relative neighborhood graphs and their relatives," Proc. of the IEEE, vol. 80, pp. 1502–1517, 1992.

[KK] Brad Karp and H. T. Kung, "GPSR: Greedy perimeter stateless routing for wireless networks", Porc. MOBICOM, August 2000, pp. 243-254.

[KMP] Goutham Karumanchi, Srinivasan Muralidharan and Ravi Prakash, "Information dissemination in partitionable mobile ad hoc networks", Proc. IEEE Symposium On Reliable Distributed Systems, Lausanne, Oct. 1999.

[KV] Y. B. Ko and N. H. Vaidya, "Location-aided routing (LAR) in mobile ad hoc networks", Proc. MOBICOM, 1998, pp. 66-75.

[LCW] X.-Y. Li, G. Calinescu, and P.-J. Wan, "Distributed construction of planar spanner and routing for ad hoc wireless networks," in 21st Annual IEEE Conference on Computer Communications (INFOCOM), 2002.

[LJCKM] Jinyang Li, John Jannotti, Douglas S. J. De Couto, David R. Karger and Robert Morris, "A scalable location service for geographic ad hoc routing", Proc. MOBICOM 2000, pp. 120-130.

[LJJCKM] Jinyang Li, John Jannotti, Douglas S. J. De Couto, David R. Karger and Robert Morris, "A Scalable Location Service for Geographic Ad Hoc Routing", Proc. ACM MOBICOM 2000, pp. 120-130.

[LJS] Dandan Liu, Xiaohua Jia and Ivan Stojmenovic, "Quorum and connected dominating sets based location service in wireless ad hoc and sensor networks", Computer Communication, 30(2007), pp. 3627-3643.

[LSW] X.-Y. Li, I. Stojmenovic, and Y. Wang, "Partial delaunay triangulation and degree limited localized bluetooth scatternet formation," IEEE Transactions on Parallel and Distributed Systems, vol. 15, no. 4, pp. 350–361, 2004.

[RKYYEGS] S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan and S. Shenker, "GHT: A Geographic Hash-table for Data-centric Storage in Sensornets", In First ACM International Workshop on Wireless Sensor Networks and Applications (WSNA), September 2002.

[S-cm2002] Ivan Stojmenovic, "Position-Based Routing in Ad Hoc Networks", IEEE Communications Magazine, Vol. 40, No. 7, July 2002, pp. 128-134.

[S-geo] Ivan Stojmenovic, "Geocasting with guaranteed delivery in sensor networks", IEEE Wireless Communications Magazine, Vol. 11, No. 6, pp. 29-37, 2004.

[S-ha] Ivan Stojmenovic, "Home agent based location update and destination search schemes in ad hoc wireless networks", in Advances in Information Science and Soft Computing (A. Zemliak and N. E. Mastorakis, editors), WSEAS Press, pp. 6-11, 2002.

[SL] I. Stojmenovic and X. Lin, "Loop-free Hybrid Single-Path/Flooding Routing Algorithms with Guaranteed Delivery for Wireless Networks," IEEE Trans. Parallel Dist. Sys., vol. 12, no. 10, 2001, pp. 1023–32.

[SLJ] Ivan Stojmenovic, Dandan Liu, and Xiaohua Jia, "A scalable quorum based location service in ad hoc and sensor networks", IEEE International Conference on Mobile Ad-hoc and Sensor Systems MASS, Vancouver, October 9-12, 2006.

[S-q1999] Ivan Stojmenovic, "A scalable quorum based location update scheme for routing in ad hoc wireless networks", Technical Report TR-99-09, SITE, University of Ottawa, September 1999.

[SRL] Ivan Stojmenovic, Anand P. Ruhil, D. K. Lobiyal, "Voronoi diagram and convex hull based geocasting and routing in wireless networks", Wireless Communications and Mobile Computing (Wiley), Vol. 6, No. 2, pp. 247-258, February 2006.

[SRLS] Juan Sanchez, Pedro Ruiz, Xuehong Liu, Ivan Stojmenovic, "Bandwidth-Efficient Geographic Multicast Routing Protocol for Wireless Sensor Networks", IEEE Sensors, Volume 7, Issue 5, May 2007, pp. 627 – 636.

[SSW] David Simplot-Ryl, Ivan Stojmenovic, and Jie Wu, "Energy efficient backbone construction, broadcasting, and area coverage in sensor networks", in: Handbook of Sensor Networks: Algorithms and Architectures (I. Stojmenovic, ed.), Wiley, 2005, 343-379.