INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

Université d'Ottawa · University of Ottawa

GPS based localized routing algorithms for wireless networks

Thesis

Submitted to the School of Graduate Studies and Research

In partial fulfillment of the requirements

For the degree of Master

December, 1999

Ottawa-Carleton Institute for Computer Science

School of Information Technology and Engineering

University of Ottawa

Ottawa, Ontario, Canada

Acknowledgement

I am very thankful to my supervisor Dr. Ivan Stojmenovic who suggested this topic for my research. He has been very patient and supportive, and provided numerous great suggestions throughout the development of this thesis.

Abstract

We discuss routing algorithms for wireless networks with the goal of achieving high (or guaranteed) delivery rate and increasing the node life in the network. Some know methods were studied: Most Forward within Radius (*MFR*) and directional algorithm (*DIR*). We propose some new location based routing algorithms: the constant metric GEographic DIstance Routing (GEDIR) algorithm and several power-aware algorithms: power efficient, cost efficient and power-cost efficient routing algorithms.

2-hop, flooding and multiple-path variants are also suggested for the algorithms with constant metric to reach a higher delivery rate while minimizing the network resource (bandwidth etc.) usage.

We will also study the quantitative metrics used to evaluate the performance of routing algorithms: delivery rate, hop count, flooding ratio, power consumption, network lifetime, etc.

Simulation experiments with static random unit graphs were designed to compare the performance of all the routing algorithms discussed. Data were collected and analyzed after each set of simulation. The study reveals that there is no clear winder, and different algorithms have their own strength in different network context.

Contents

List of Figures, Tables, Graphs and Images

Figures

Tables

Graphs

Images

Chapter 1. Introduction

In this chapter, we introduce the Mobile Ad hoc Network (MANET). Then we represent some routing algorithms in MANET, including known and new Global Positioning System (GPS) based routing algorithms.

1.1 Introduction to MANET

In this thesis, we consider the routing task, in which a message is to be sent from a source node to a destination node (in a sensor or ad hoc wireless network). The nodes in the network may be static (e.g. thrown from an aircraft to a remote terrain or a toxic environment), static most of the time (e.g. books, projectors, furniture, motors) or moving (vehicles, people, small robotic devices).

Wireless networks of sensors are likely to be widely deployed in the near future because they greatly extend our ability to monitor and control the physical environment from remote locations and improve our accuracy of information obtained via collaboration among sensor nodes and online information processing at those nodes. Networking these sensors (empowering them with the ability to coordinate amongst them on a larger sensing task) will revolutionize information gathering and processing in many situations. Sensor networks have been recently studied in [BBCDFLMM, BM, EGHK, FFP, HCB, HKB, KKP]. A similar wireless network that received significant attention in recent years is mobile ad hoc network [IETF, MC].

Mobile ad hoc networks (MANET) consist of wireless hosts that communicate with each other in the absence of a fixed infrastructure. Routes between two hosts in MANET may consist of hops through other hosts in the network. Some examples of the possible uses of ad hoc networking include soldiers on the battlefield, emergency disaster relief personnel, network of laptops at conferences and meetings etc. The task of finding and maintaining routes in MANET is nontrivial

since host mobility causes frequent unpredictable topological changes. A number of MANET protocols for achieving efficient routing have been recently proposed. They differ in the approach used for searching a new route and/or modifying a known route, when hosts move. The surveys of these protocols, that do not use geographic location in the routing decisions, are given in [BMJHJ, RS].

1.2 Routing algorithms in MANET

Global Position System (GPS) provides location information (latitude, longitude and possibly height) to mobile users. In mobile ad hoc wireless network, hosts equipped with a device to take advantage of GPS can obtain the location information of its neighboring hosts and that of the destination. This information can help hosts in the selection of an appropriate routing path. Another benefit of GPS is that it gives the hosts in the MANET the ability to adjust the transmission power to the minimal level required for successful transmission/reception.

The benefit of GPS in MANET is obvious and immediate: instead of sending requests between hosts in the network to locate the destination host, now the sending node can obtain that information via GPS immediately. This saves time and scarce network bandwidth, and extends the lifetime of hosts in the network. In accordance to several studies [BCSW, KV, KSU] it was shown that GPS based routing methods perform significantly better than the methods that do not use geographic location in routing tables. In the thesis we will discuss only the GPS based approaches and how to efficiently perform message routing in an ad hoc network environment while keeping the energy consumption level at a minimum. More detail will be discussed in the next chapter.

1.3 Assumptions

Although 'algorithm' and 'protocol' have the same meaning in literature, they do have a subtle difference in our discussions. The routing methods are described by algorithms which underline only major ideas of the corresponding detailed protocol. The actual protocol may always include additional techniques, most of them already being applied in other protocols, and details of communication between nodes. This thesis will focus on routing algorithms, not protocols.

A number of novels on routing protocols in MANET had been published and are available on the Internet [IETF]. However, most existing routing algorithms do not consider the power consumption in their routing decisions. Usually shortest weighted path algorithms would achieve the best performance in network routing. However, all these routing methods require the knowledge of the locations of all nodes in the network, and the information about the existence of every edge in the graph. It would outperform any other routing protocols in a static network, but in an ad hoc network where nodes change location in an unpredictable way continuously, such algorithms may not be the best choice: constant update of network table is required whenever a host moves, or nodes in "sleeping" status (inactive) could be in the routing path discovered, and thus causes the routing path to be broken, resulting the need to rediscover a new path.

While the absence of a GPS receiver at nodes forces many problems (routing, for example) to be solved by only non-localized algorithms, their application, when GPS is available, may not be justified if efficient localized alternatives are available. The power savings obtained by applying a localized rather than non-localized algorithm outweighs even the additional power imposed to nodes for carrying and using GPS receiver.

In accordance with [BCSW, BMSU, KV, KSU, LS, SL1, TK], it is assumed that each node, in its internal routing table, contains the geographic location of all other nodes in the network,

including the time when the location for each node is established. Since nodes may move, the actual location may differ from the one recorded in the routing table, and, for each node, its location information in other nodes may be different. However, the routing table (in localized algorithms) is only used to provide approximate position of destination, and accurate information about location of neighboring nodes.

Furthermore, it is also assumed here that each node is aware of its inactive neighbors (and possibly inactive 2-hop neighbors). The algorithms discussed in this thesis use only the location of destination and location (and activity) information of direct neighbors (and possibly 2-hop neighbors) to make a decision on forwarding the message (in distributed manner). In 1-hop and 2-hop methods, there is exactly one copy of each message in MANET at all times, that is, each intermediate node will forward the message to exactly one of its neighbors. The memory requirements for storing the information about the past traffic at each node differ in algorithms that will be discussed. In 1-hop and 2-hop algorithms, nodes do not memorize any message previously forwarded to any of neighbors. Messages in flooding based algorithms or multiple path methods memorize past traffic at each node to prevent high flooding ratio.

1.4 Known GPS based routing algorithms

The study of GPS based routing methods can date back to mid 80's, but it was recently that it receives more focus. Several new GPS based algorithms were introduced to the community [BCSW, KV, KSU, HL, NK, TK]. However, it is unlikely to expect that one routing protocol for MANET is the best approach for all networking contexts. Thus, it is not surprising to find a number of hybrid methods in literature, combining several major ideas into one single protocol. In chapter 2, we shall review existing routing protocols [BCSW, KV, KSU, HL, NK, TK] which use

geographic location in their route decisions, and some qualitative and quantitative metrics used for judging the performance of the routing protocols. Although [BCSW] claims that directional routing algorithms (*DIR*) provide loop-free paths, we shall give a counterexample showing that undetected loops can be created. Our literature review revealed some other GPS based methods from early 1980's [HL, NK, TK]. One of them, Most Forward within Radius (*MFR)* method [TK] is a competitive method and we prove here that it is loop-free.

1.5 Contributions

We shall describe the Geographic Distance Routing (*GEDIR*) algorithm, and prove that it is inherently loop-free. The proof does not use unit graph properties, and is therefore valid for any metric network that satisfies the triangle inequality, including networks in three-dimensional spaces. Several modifications to *GEDIR, MFR* and *DIR* methods, which should provide a better trade-off between delivery and flooding rates, are also described here. Neighbors of neighbor (2-hop neighbors) may be used to enhance delivery rate and shorten hop count. Flooding may be used at nodes where basic method drops the packet. Finally, the sender may initialize c paths toward destination, to provide multiple paths that involve significantly lower flooding rates.

The *GEDIR* algorithm uses the geographical information of its neighbor hosts and destination node to determine the recipient of the message packet: the neighbor node that is geographically closest to the destination node. The implementation is straightforward, and it only uses few CPU cycles: two subtractions, two multiplications and one addition per neighboring node, and n comparisons for all its neighbors to reach the verdict. Full text on this algorithm is available at chapter 3.

Then we will describe several other power-oriented algorithms in chapter 4. Power aware localized routing algorithm and Nearest-Closer *(NC)* attempt to minimize the total power needed to route a message between a source and a destination. Cost-aware localized algorithm is aimed at extending battery's worst case lifetime. The combined power-cost localized routing algorithm tries to minimize the total power needed and to avoid nodes with short battery's remaining lifetime. Since most of proposed GPS based algorithms do not consider the power consumption in their routing decisions, thus the power aware algorithms have a big advantage over other algorithms in saving energy and extending lifetime of hosts in the network.

The two joint papers about the above contributions (one for *GEDIR* [SL1] and another one for *power-aware* [SL2]) had both been accepted for publication in conferences and the first one *(GEDIR)* [SL1] is actually published.

1.6 Analysis

The routing protocols designed in literature are, in most cases, evaluated by using a discrete event simulator on certain kind of graphs, with particular parameter values (e.g. topological rate of change, various traffic patterns, mobility patterns, fraction and frequency of sleeping nodes [MC]). While such evaluation is an ultimate goal for GPS based routing protocols, the scope of our research is to study candidate GPS based routing algorithms that will serve as a basis for the design of routing protocols. In the presence of a number of possible algorithms that we proposed, the performance evaluation should begin with the case of static nodes, for which routing does not require control messages. After the best algorithms are filtered, each of them may be combined with few different methods for sending control messages to determine the best GPS based protocol.

To better evaluate the strength of those GPS based routing protocols, several experiments with different emphasis were designed and performed, and different independent quantitative metrics were collected during the experiments. Comparisons were made and conclusions were drawn based on the simulation results. An event simulator was developed to simulate different algorithms' behaviors in their routing tasks in a static network. Another software package was also written to help examine more closely the path discovered by each method. We had elected to use a static network for the goal to first have some in-depth analysis on the capacity and weakness of these algorithms, filter out some, and then focus our attention to the rest for further study. Detailed analyses are given out in chapter 5, 6 and 7.

It is unlikely to expect that one routing protocol for MANET is the best approach for all networking contexts. According to [MC], parameters that define a networking context, in case of static network with nodes of equal range and capacity, are network size n (the number of nodes), and network degree (i.e. connectivity) d. Our experiments were designed to compare all methods in terms of their average delivery rates, hop counts, flooding rates, and power consumption and network lifetime. The Dijkstra's shortest weighted path algorithms *(SP)* were used as benchmark (and Breath First algorithm is used to test whether a graph is connected).

Chapter 2. Routing algorithms in MANET

In this chapter, we shall give more in-depth reviews on routing protocols in MANET. Some known quantitative metrics were studied and known GPS based localized routing algorithms were discussed.

2.1 Role of GPS in routing algorithms in MANET

In the near future, a broad variety of location dependent services will become feasible due to the use of the Global Position System (GPS), which provides geographical location information (latitude, longitude and possibly height) and global timing to mobile users. GPS cards will be deployed in each car and possibly in every user terminal [K, NI]. For instance, NAVSTAR Global Positioning System has a potential accuracy of about 50-100 meters and Differential GPS offers accuracy of a few meters [N]. In the USA, Federal Communications Commission has adopted rules requiring wireless service providers to supply two-dimensional location information of mobile users who request the E-911 emergency service [EMMB]. Navas and Imielinski [NI] described GPS's application in geographic messaging to users who are located within a particular polygon or circle defined by latitude and longitude. Their method is based on a hierarchy of geographically defined routers, and the intersection of the appropriate levels of routers with the given polygon or circle.

Mobile Ad Hoc Networking (a.k.a. Mobile Packet Ratio Networking) is a name currently being given to a technology under development for the past 20 or so years. A Mobile Ad hoc NETwork (MANET) consists of mobile hosts (which are free to move about arbitrarily) that communicate with each other, in the absence of a fixed infrastructure. Routes (communication) between two hosts in a Mobile Ad hoc Network may consist of hops (transmission between hosts) through other

hosts in the network. As the words 'Mobile Ad hoc' suggest, hosts in a MANET can move at anytime to any direction, thus causing frequent unpredictable topology changes to the network. Therefore, the task of finding and maintaining routes in MANET is non-trivial.

In real world, electric power is consumed when nodes communicate between each other. While the computational power of the devices used in MANETs is rapidly increasing, the lifetime of batteries is not expected to improve much in the future. The average life of batteries in an idle cellular phone is one day [MG]. The DEC Roamabout radio [h] consumes approximately 5.76 watts during transmission, 2.88 watts during reception and 0.35 watts when idle. The radio used in [GFM] consumes 15 watts while transmitting, 11 watts while receiving and 50mW in idle mode. Low power displays, I/O devices, CPU's and algorithms to reduce power consumption of disk drives are being developed [SWR]. We see a clear need for improvement in the power consumption in existing MAC protocols and routing algorithms: in almost all of the current protocols, nodes are powered on most of the time even when they are doing no useful work [SC, SR, SWR]. Power-saving asynchronous communication protocols were discussed in [CZ, SC, SR].

Global Position System (GPS) provides location information (latitude, longitude and possibly height) to hosts in ad hoc wireless network. The location information allows each node to adjust the transmission power to the minimum needed for successful reception. The recent low-power implementation of a GPS receiver [SSL] makes its presence a viable option in minimum energy network design.

To accurately model the attenuation of radio waves between antennas close to the ground, radio engineers typically use a model that attenuates the power of signal as $1/r^2$ at short distances (free space propagation model), and as $1/r^4$ at longer distances (two-ray ground reflection model), where r is the distance between antennas. The crossover point is called the reference distance, and is

typically around 100 meters for outdoor low-gain antennas 1.5m above the ground plane operating in the 1-2GHz band [R]. In order for a message to be received correctly, the power of the signal should be above reception threshold. According to [PL], the power of signal received at a node is inversely proportional to the distance the receiver is from the transmitter, raised to an exponent known as the distance-power gradient, i.e. $P=T/d^{\alpha}$, where P is power of the received signal, T is the power of the transmitted signal, d is the distance between the receiver and the transmitter, and α is the distance-power gradient. In an ideal situation $\alpha=2$. However, due to various environmental factors such as building materials, street layouts, terrain characteristics etc., the measured value of α may vary from less than two to more than six.

Several routing algorithms [HL, NK, TK] have used signal attenuation to determine probabilities of capturing transmitted signals, and proposed techniques to increase that probability. In order to reduce the negative impact of collisions, [CC] proposed the use of directional antennas (the packet is transmitted from the source within a particular angular range) instead of omnidirectional ones (the packet is transmitted to all nodes within a circle of given radius). In [KKKP], the signal attenuation was used to study the problem of assigning transmission ranges to the nodes of a multi-hop packet radio network as to minimize the total power consumed under the constraint that adequate power is provided to the nodes to ensure that the network is strongly connected. It is proved in [GK] that n nodes placed in a disk of unit area define connected unit graph of radius R with probability one if and only if $R> (\log n)/n$, when n approaches infinity. Various aspects of power management for wireless communication networks are reviewed in [RB].

We shall also use several power-related metrics, which will be discussed, in the coming sections.

Ad hoc networks are best modeled by *minpower* graphs constructed in the following way: each node A has its transmission range $t(A)$. Two nodes A and B in the network are neighbors (and thus joined by an edge) if the Euclidean distance between their coordinates in the network is less than the minimum between their transmission radii (i.e. $d(A, B) < min\ \{t(A), t(B)\}$) [BCSW]. If all transmission ranges are equal (to the radius R of the graph), the corresponding graph is known as the *unit graph*. These models provide acknowledgments for received messages. The minpower and unit graphs are valid models when there are no obstacles in the signal path (e.g. a building). Ad hoc networks with obstacles can be modeled by subgraphs of minpower or unit graphs. This thesis deals primarily with unit graphs.

2.2 Localized vs. Non-localized

Ad hoc networks consist of autonomous nodes that run their routines in asynchronous fashion. The communication algorithms between nodes are therefore all distributed. However, Estrin, Govindan, Heidemann, and Kumar [EGHK] defined the class of so called *localized* algorithms, as distributed algorithms where simple local node behavior achieves a desired global objective. Localized algorithms therefore resemble the class of greedy sequential algorithms. They argued that localized algorithms may be necessary for sensor and ad hoc network coordination, and described localized algorithms for clustering and object location. We propose a more formal definition of localized algorithms. In a localized algorithm, each node makes the decision to which neighbor(s) to forward the message based solely on the location of itself, its neighboring nodes, and a *constant* amount of additional information. In addition, each node is allowed to perform local computation (since it is equipped with a small but powerful processor). The decision depends on the task performed and the algorithm followed, which is part of an incoming message. In case of

routing, the additional information needed is the location of the destination. The sender uses the latest available location about the position of the destination and attaches it to the message. Intermediate nodes may use their more recent location information to replace the one that comes with the message (and to update their own internal table if necessary). Note that the requirement for the location of a destination means that each node should update locations of all other nodes in the network, so that any routing that is initiated will have a reasonably good start. However, the path adjustments can be made as the message travels closer to the destination.

Non-localized algorithms, on the other hand, typically require the knowledge of the locations of all nodes in the network, and the information about the existence of every edge in the graph. All non-localized routing algorithms proposed in literature are variations of shortest weighted path algorithm (e.g. [CN, LL, RM, SWR]). Although ad hoc network is fairly accurately modeled by unit graphs, nodes that are at distance less than R may have an obstacle between them blocking the communication. On the other hand, two nodes at distance that exceeds R by a small amount may still be able to communicate between them (or a node may even choose whether to use that possible but power demanding link). If any edge in the network breaks or emerges (due to node mobility); it may have impact in the whole network. In the worst case scenario, a path that used to be shortest may not even exist, and the algorithm does not offer immediately a reasonable alternative path unless the shortest path algorithm is rerun. Therefore the maintenance of shortest weighted path requires that the information about edges, in addition to location of nodes, is broadcast to the whole network, which is a significant quadratic communication overhead, and may cause significant delays in delivery (before new paths are available, shortest weighted paths need to be recalculated). Next, some nodes in the sensor or ad hoc network may be temporarily inactive, and non-localized algorithms need to know which of the nodes are active to make their best decisions. The activity

information puts additional demand on the information update. For example, static nodes may need to broadcast such information to the whole network whenever they change their activity status while, at the same time, they have no need to update their location with the rest of the nodes. In order to preserve battery power over long period, nodes may change their activity on a regular basis, or it may depend on the size of the job assigned to them. Thus, non-localized shortest path algorithms may not be the best choice for a routing algorithm even in case of static networks (e.g. some kinds of sensor networks). While the absence of a GPS receiver at nodes forces many problems (routing, for example) to be solved by only non-localized algorithms, their application, when GPS is available, may not be justified if efficient localized alternatives are available. The power savings obtained by applying a localized rather than non-localized algorithm outweighs even the additional power imposed to nodes for carrying and using GPS receiver. This thesis deals solely with localized algorithms.

2.3 Known quantitative metrics and routing algorithms

To compare performance of different routing algorithms, different measurements (metrics) were used. Here we shall discuss known metrics and some known routing algorithms.

2.3.1 Constant metrics

Macker and Corson [MC] listed qualitative and quantitative independent metrics for judging the performance of delivery-oriented routing protocols. Desirable qualitative properties include distributed operation, loop-freedom (to avoid a worst case scenario of a small fraction of packets spinning around in the network), demand-based operation, and 'sleep' period operation (when some nodes become temporarily inactive). Some quantitative metrics that are appropriate for assessing

the performance of any routing protocol include [MC]: end-to-end data delay, average number of data bits (or control bits) transmitted per data bits delivered. In this thesis we use three quantitative metrics that are similar to these described in [BMJHJ] (each of them being an average value):

- *hop count* (the number of edges, i.e. transmissions on the path from source to destination),
- *delivery rate* (the ratio between numbers of messages arrived at destination and sent by senders),
- *flooding rate* (the ratio of the total number of transmissions over the shortest possible hop count between two nodes). Each transmission in multiple routes is counted, and message can be sent to all neighbors with one transmission.

We would refer this metrics as constant metrics because the measurement of this metrics is constant no matter how far two nodes are away from each other. For example, a node currently holding the message is forwarding the message to its neighbor node, and this would count as one hop count, one transmission, no matter how far this neighbor is away from the current node.

Guaranteed delivery is always desirable, and a message delivered with minimum hop counts is even better. It is a given fact that all the algorithms in the literature have that aim at maximizing the delivery rate, so hop count is often used to evaluate the performance of those localized algorithms such as *MFR, DIR, DREAM, LAR* etc. [TK, KSU, BCSW, KV]. On the other hand, the most popular non-localized algorithm that would achieve the optimal value for that metrics is the Dijkstra's shortest weighted path algorithm *(SP)* where each edge is equally weighted to a constant value (usually value 1).

2.3.2 Transmission cost/reluctance

In most of routing protocols, the paths are computed based on minimizing hop count or delay. Some nodes participate in routing packets for many source-destination pairs, and the increased energy consumption may result in their failure. For example, hierarchical and spine routing algorithms will (by their own design) exploit nodes that lie on the spine in order to reduce message overhead in routing table maintenance. This metric of reducing message overhead may be misguided in the long term, as observed in [SWR]. A longer path that passes through nodes that have plenty of energy may be a better solution [SWR].

The algorithm [SWR] proposes to use a function $f(A)$ to denote node A's reluctance to forward packets, and to choose a path that minimizes the sum of $f(A)$ for nodes on the path. This routing protocol [SWR] addresses the issue of energy conservation in critical nodes. As a particular choice for f, [SWR] proposes $f(A)=1/g(A)$ where $g(A)$ denotes the remaining lifetime, normalized to be in the interval [0,1]. Thus, reluctance grows significantly when lifetime approaches zero. We would refer such metric as the *transmission-cost* metric because it demonstrates how much it would "cost" the node to retransmit the message. Another metric used in [SWR] is aimed at minimizing the total energy consumed per packet, and further discussion will be given at the following section. However, [SWR] merely observes that the routes selected when using this metric will be identical to routes selected by shortest hop routing, since the energy consumed in transmitting (and receiving) one packet over one hop is considered *constant.* In other words, the algorithm [SWR] that minimizes the total energy consumption is equivalent to the shortest hop count, since it is assumed that each transmission requires equal power. For each of the two proposed power consumption metrics (cost and hop count), [SWR] assigns weights to nodes or edges, and then

refers to non-localized Dijkstra's algorithm for computing shortest weighted path between source and destination. Thus for this *cost* function, the weight of each edge in Dijkstra's *shortest-path (SP)* algorithm is assigned with its *cost* (reluctance) to forward the message. We also observed that the validation of power aware metrics in [SWR] was not done on minpower graphs but on random graphs where each pair of nodes is joined by an edge with a fixed probability p.

2.3.3 Power consumption

Recall that in previous section 2.1, we have discussed briefly methods used to calculate power consumption. In this section, we shall give a more thorough study on this *power* metric. Let P and T be the powers of received and transmitted signals, respectively. The well known relation states that $P=T/d^{\alpha}$, where d is distance between sender and receiver nodes, and α is a constant [R, PL]. We first observe that the expression is somewhat imprecise. First of all, if d is expressed in absolute measure (say, in meters), then the two sides of the equation do not have the same physical meaning (that is, if the left side is in watts then the right side is in watts per meter). Therefore d is a relative unit, that is, the exact distance between nodes is divided by some unit distance. Next, we observe that, for $d<1$, the received power is greater than transmitted one, which is physically impossible. Thus, this model was rejected in favor of models described below.

In the simple radio model [HCB], radio dissipates E_{elec}=50 nJ/bit to run the transceiver circuitry. Both sender and receiver node consume E_{elec} to transmit one bit between them. Assuming d^2 energy loss, where d is the distance between nodes, transmit amplifier at the sender node consumes further $E_{amp}d^2$, where E_{amp} = 100 pJ/bit/m^2. Thus, to transmit one bit message at distance d, the radio expends $E_{elec} + E_{amp}d^2$, and to receive the message, the radio expends E_{elec}. In order to normalize the constants, divide both expressions by E_{amp}, so that radio expands $T=E + d^2$

for transmission and $P=E$ for reception, where $E= E_{elec} /E_{amp}$=500 m^2. Note that $T/P= 1+ d^2/E$ and $T+P=2E+ d^2$. Therefore, in this model, referred to as model 2, the power needed for transmission and reception at distance d is $u(d)= 2E+ d^2$.

Rodoplu and Meng [RM] considered, in their experiments, the model with $u(d)= d^4+2*10^8$. They also study a general formula $u(d)=d^{\alpha}+c$. In order to present some properties in a uniform way, let us assume that the power needed for transmission and reception between two nodes at distance d is $u(d)=ad^2 + bd +c$. In the first model, $a=1$, $b=2$, $c= e+1$, while in the second model $a=1$, $b=0$, $c=2E$. Rodoplu and Meng [RM] adopted also a general model in which the power consumption between two nodes at distance d is $d^{\alpha}+c$ for some constants α and c, and described several properties of power transmission that are used to find neighbors for which direct transmission is the best choice in terms of power consumption. They also described several properties of power transmission that are used to find neighbors for which direct transmission is the best choice in terms of power consumption. They discussed that large-scale variations (modeled by lognormal shadowing model) can be incorporated into the path loss formula, and that small-scale variations (modeled by a Rayleigh distribution) may be handled by diversity techniques and combined at the physical layer.

Rodoplu and Meng [RM] described a power aware routing algorithm that runs in two phases. In the first phase, each node searches for its neighbors and selects the neighbors for which direct transmission requires less power than if an intermediate node is used to retransmit the message. This defines so-called *enclosure* graph. In the second phase, each possible destination runs distributed loop-free variant of (non-localized) Bellman-Ford shortest path algorithm and computes shortest path for each possible source. The same algorithm is run from each possible destination. We observed that since the energy required to transmit from node A to node B is the same as energy

needed to transmit from node *B* to node *A*, the same algorithm may be applied from each possible source and used to discover the best possible route to each destination node. Alternatively, alternatively, it may be used to find the location of destination and the best route to it.

Heizelman, Chandrakasan and Balakrishnan [HCB] used signal attenuation to design an energy efficient routing protocol for wireless microsensor networks, where destination is fixed and known to all nodes. They propose to utilize a 2-level hierarchy of forwarding nodes, where sensors form clusters and elect a random clusterhead. The clusterhead forwards transmissions from each sensor within its own cluster. This scheme is shown to save energy under some conditions. However, the scheme does not apply to our case since destination is not fixed in our routing problem. Nevertheless, their simple radio model and metric will be considered in our thesis, and will be discussed in chapter 4.

2.4 Known GPS based localized routing methods

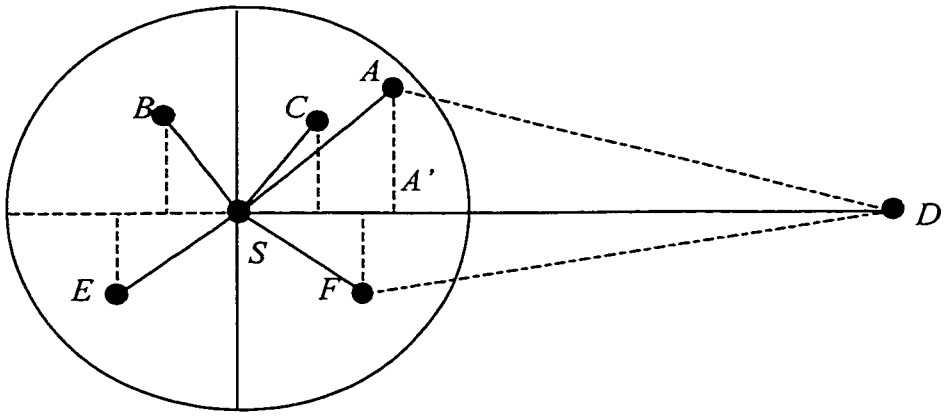


Figure 2-1. Progress based routing methods (MFR)

Most existing routing algorithms do not consider the power consumption in their routing decisions. Instead, their goal is to achieve maximum delivery rate with minimum hop counts. We should refer those algorithms as algorithms with constant metrics in the future.

2.4.1 Progress based routing algorithms

Several GPS based methods were proposed in 1984-86 by using the notion of progress. Progress is defined as the distance between the transmitting node and receiving node projected onto a line drawn from transmitter toward the final destination. A neighbor is in forward direction if the progress is positive (for example, for transmitting node S and receiving nodes A, C and F in Figure 2-1); otherwise it is said to be in backward direction (e.g. nodes B and E in Figure 2-1). In the random progress method [NK], packets destined toward D are routed with equal probability towards one intermediate-neighboring node that has positive progress. The rationale for the method is that, if all nodes are sending packets frequently, probability of collision grows with the distance between nodes (assuming that the transmission power is adjusted to the minimum required), and thus there is a trade-off between the progress and transmission success. In [HL], packet is sent to the nearest neighboring node with forward progress (for instance, to node C in Figure 2-1).

Recently, several routing algorithms that use geographic distance were proposed. In [BBCDFPMM, BM], an online routing algorithm was studied in geometric graphs and triangulations. In 1999, a greedy algorithm that utilizes geographic distance was presented [FFP]. It was shown that such greedy algorithm was optimal for some simple topologies (e.g. some chordal grids). Formation of local loops was not allowed in this algorithm, and a sufficient condition was given to prevent bigger loops from being formed. These above two algorithms [BBCDFPMM, BM, FFP] are similar to the algorithm (*GEDIR*) that we have proposed for unit graphs in [LS, SL1].

Takagi and Kleinrock [TK] proposed Most Forward within Radius (*MFR*) routing algorithm, in which packet is sent to the neighbor with the greatest progress (e.g. node A in Figure 2-1). In [HL], this method is modified by proposing to adjust the transmission power to the distance between the two nodes. We shall reformulate the *MFR* method in order to facilitate its implementation and provide a simple proof that it is loop-free. Let $a \bullet b$ denote the dot products of vectors a and b. Consider the dot products of vectors originating from destination D and ending at nodes in MANET. Clearly $DS \bullet DA = |DS||DA'|$ where A' is the projection of A on the line DA (see Figure 2-1). The sign is assumed here to be positive; it can be shown that, in case of negative dot product, D must be a neighbor of S. Thus the considered dot product is exactly minimal when the progress in maximal. The goal in the *MFR* algorithm [TK] is, therefore, to minimize the dot product. Note that the node that minimizes the dot product (the selected neighbor) may not have a forward progress. So we will first consider among positive progress neighbors and choose the node that has the minimal dot product. If no positive progress neighbor exists, then the node that has maximal (negative) dot product. Using the dot product definition, we shall prove, in the next section, that the *MFR* algorithm is loop-free.

2.4.2 Direction based routing algorithms

Recently, three articles [BCSW, KV, KSU] independently reported variations of fully distributed routing protocols based on direction of destination. In these directional routing methods, node A uses the location information for B and its one hop neighbors to obtain B's direction, and then transmits a message m to several neighbors whose direction (looking from A) is closest to the direction of D. The methods differ in the choice of direction ranges.

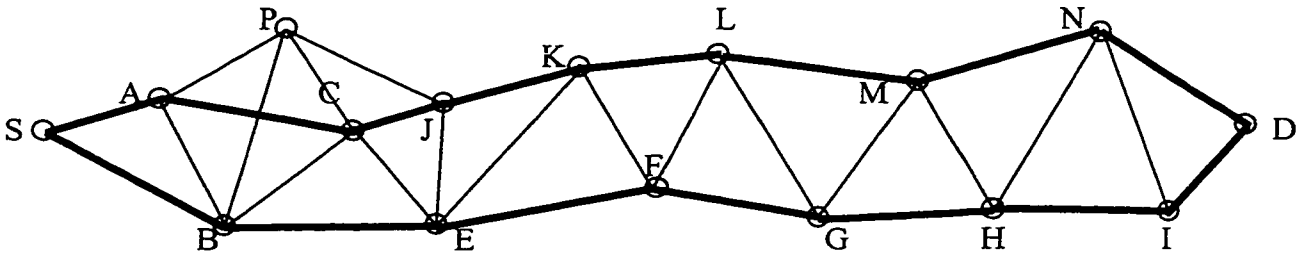


Figure 2-2. Paths selected by *DIR (SACJKLMND)* and *GEDIR (SBEFGHID)* algorithms

In the *compass routing* method (referred here also as the *DIR* method) proposed by Kranakis, Singh and Urrutia [KSU], the source or intermediate node *A* uses the location information for the destination *D* to calculate its direction. The location of one hop neighbors of *A* is used to determine for which of them, say *C*, has the direction *AC* closest to the direction of *AD* (that is, the angle *CAD* is minimized). The message *m* is forwarded to *C*. This process repeats until the destination is, hopefully, reached. Consider MANET on Figure 2-2, where the radius is equal to edge *EF*. The direction *AC* is closest to direction *AD* among candidate directions *AS, AB, AC,* and *AP*. The path selected by *DIR* method is *SACJKLMND* and consists of eight hops. Although the authors describe their method for static networks only (for finding routes using only compass and the geographic road maps), it may be used also in ad hoc networks. They gave a counterexample showing that the compass routing is not loop-free even for static networks modeled by planar graphs embedded in plane (geometric graphs, which differ from unit graphs). The authors modify their algorithm to avoid loops and guarantee delivery for the special case of planar graphs with convex regions and few other cases, which do not correspond to realistic ad hoc networks.

Basagni, Chlamtac, Syrotiuk and Woodward [BCSW] described a distance routing effect algorithm for mobility (DREAM). The source or any intermediate node *A* calculates the direction of destination *D* and, based on the mobility information about *D*, chooses an angular range. The

message *m* is forwarded to all neighbors whose direction belongs to the selected range. The range is determined by the tangents from *A* to the circle centered at *D* and with radius equal to a maximal possible movement of *D* since the last location update. The area containing the circle and two tangents is referred as the request zone in [KV]. DREAM algorithm [BCSW] incorporates the idea of triggering the sending of location updates by the moving nodes autonomously at a rate and hop distance that correspond to the node's mobility rate. Ko and Vaidya [KV] described, independently at the same conference, an almost identical algorithm, and a few modifications of it. In the location aided routing *(LAR)* algorithm [KV], the request zone is fixed from the source, and a node that is not in the request zone does not forward a route request to its neighbors. If the source has no neighbors within the request zone, the zone is expanded to include some. The size of the request zone depends on the average speed of the destination's movement and time elapsed since the last known location of the destination was recorded [BCSW, KV].

The definition of the request zone [BCSW, KV] was modified in [LS] in order to provide uniform framework with the corresponding notions in *GEDIR* and *MFR* methods. [LS] discussed the *V-GEDIR, CH-MFR* and *R-DIR* methods; in which *m* is forwarded to exactly those neighbors which may be best choices for a possible position of destination (using the appropriate criterion). The request zone in *R-DIR* method [LS] may include one or two neighbors that are outside of angular range, because they can have the closest direction for the tangents to the circle. In *V-GEDIR* method, these neighbors are determined by intersecting the Voronoi diagram of neighbors with the circle (or rectangle) of possible positions of destination, while the portion of the convex hull of neighboring nodes is analogously used in the *CH-MFR* method.

Ko and Vaidya [KV] discussed various enhancements to their basic technique. The *LAR scheme 1* [KV] proposes an alternative definition of the request zone, as the smallest rectangle that

includes current location of S and the expected zone of destination (a circular region). The request zone is thus increased, with increased chances of reaching destination but also with increased flooding. The modifications in [KV] include sending route requests before the message itself [JM]. Note that a route request may be considered as a routing of short messages. Nodes may update their location information with each exchange of messages between them. Messages may also contain source location to update location information at intermediate nodes. Recovery procedures based on partial or full flooding, to start flooding if the given algorithm fails to find the route within a timeout interval, are proposed by both papers [BCSW, KV].

Ko and Vaidya [KV] also proposed the *LAR scheme 2*. In this scheme, the source or each intermediate node A will forward the message to all nodes that are closer to the destination than A is (more precisely, at most δ further from the destination than node A, to account for possible location error). This scheme therefore suggests the use of geographic distance instead of direction.

The routing algorithms in [BCSW, KV] are fully distributed, and robust, since they provide multiple routes. They are also demand-based and adapt well to 'sleep' period operation. Simulation results presented in [BCSW] using a discrete event simulator show that the dynamic source routing protocol [JM] has a 25% to 250% larger end-to-end delay than the *DREAM* protocol. The average number of data bits transmitted per data bits delivered is consistently lower for both *LAR* schemes as compared to flooding [KV]. Therefore adding location information to the routing tables in all nodes resulted in significant improvement in the performance over the existing methods that do not use such information. Despite these advantages, the proposed methods [BCSW, KV] have some drawbacks. They have considerable flooding rates, and the directional methods are shown (in this thesis) not to guarantee loop-free paths. This thesis attempts to improve on these two measures.

In [CL], routing tables are used and they are updated by mobile software agents modeled on ants. Ants are used to collect and disseminate information about nodes' location.

Chapter 3. New GPS based localized constant metrics routing algorithm

In this chapter we shall discuss an important routing property: *loop-freedom*, and prove that directional routing algorithm is not loop-free, but progress based algorithm is. Then we shall propose a new GPS based fully distributed localized constant metric routing algorithm: *GEDIR*, and prove that this algorithm is indeed loop-free. At the end of the chapter we will further explore some variants of *GEDIR* and other know basic GPS based localized routing algorithms.

3.1 Loop-freedom property

Sometimes overlooked, but an important and desirable quality, *loop-freedom* is the property which message will NOT be circulated within a local loop indefinitely. Any routing algorithms without such property would have low delivery rate and network would gradually be crowded with looped messages that never reach their destination. In the following sections, three basic algorithms: *DIR* (directional routing), *MFR* (progress based method) and *GEDIR* (our new progress based routing algorithm) were explored for the *loop-freedom* property.

In order to provide uniform and fair treatment of all three basic algorithms, we assume that the message is dropped at *concave node*. An intermediate node A is a *concave node*, if the node C, selected for forwarding by A using the corresponding algorithm, is exactly the node which sent the message to A in the previous step, thus forming a loop. Such node A will be referred to as the *concave* node (in each of corresponding methods), and the loop formed with this *concave node* will be referred as *local-loop*. Concave node A in *GEDIR* algorithm is therefore a node which is closer to destination D than any of its neighbors, and node C, the closest to D among A's neighbors, has itself no closer (to D) neighbor than A. Similar definitions, using direction or dot product instead of

distance, for corresponding concave node in the compass (i.e. *DIR*) or *MFR* routing algorithms can be given. Thus, the stoppage criterion is the same for all three basic algorithms.

3.1.1 Loops in directional routing methods

The authors [BCSW] claim that their algorithm provides loop-free paths (no proof was given). However, Figure 3-1 shows a counterexample of a loop that consists of 16 nodes (thus this is NOT a local-loop), denoted A_1 to A_{16}, positioned at two close circles centered at the destination D (approximately located at nodes of two regular octagons). The graph is a unit graph with the radius equal to the length of longer edge e.g. A_1A_2 in the loop. Let the source be any node in the loop, e.g. A_1. Node A_i selects node A_{i+1}, i=1,2,3,...,16, to forward the message, because the direction of A_{i+1} is closer to D than the direction of its other neighbor A_{i-1} ($A_{17}=A_1$, $A_0=A_{16}$). Additional node C can be taken just outside the polygon defined by the loop, near the middle of the larger side e.g. A_5A_6 of the 16-gon. It can be verified that there exists a nonempty region inside the 16-gon (loop), reachable from C but not reachable from any other points on the loop. Any node B inside that region can be reached from C and is able to reach the destination. Node C can be selected such that the node A_8 has closer direction to D than C, measured from node A_7 (thus A_7 forwards message to A_8, not to C). So starting from node A_1, the message is transmitted to A_2, A_3 and to A_4. Then at node A_4 since the direction of A_5 is closer to D than the direction of node C, so A_4 picks A_5 as its next recipient. Since node C is outside the polygon, so A_6 is chosen to receive the message from A_5. Because the direction of node A_7 is closer to D than that of node A_5, so obviously node A_7 gets the message, and A_7 forwards message to A_8, as explained above. Now, the transmission continues from A_8 to A_9 until A_{15} and back to A_1, so a loop is formed. This example shows that the loop

(indicated by arrows) can be created non-locally, and with static nodes. The nodes on the loop are not able to recognize the loop unless message *id* is memorized (for each forwarded message!).

The example in Figure 3-1 is not restricted to the unit graph model of MANET. Clearly, such example may exist in any kind of random network model (model where each edge is selected with certain nonzero probability), in a subgraph of unit graph that models MANET with obstacles, or in any model that generalizes unit graphs (e.g. minpower graphs), or in any graph model that includes unit graphs as subgraphs. Finally, static network is special case of a moving network, so the counterexample is valid for ad hoc networks. Thus, by counter-example, we have shown that:

Any routing algorithm for ad hoc wireless networks in which a node currently holding the message forwards it to its neighbor with closest direction toward destination (and to some other nodes) is not a loop-free algorithm.

3.1.2 Loop-freedom in MFR

We shall now prove that the *MFR* algorithm [TK] is loop-free. Suppose that, on the contrary, there exists a loop which is NOT local (no *concave node* exists in the loop) in the algorithm. Let $A_1, A_2, \ldots A_n$ be the nodes in the loop, so that A_1 sends the message to A_2, A_2 sends the message to $A_3, \ldots, A_{n-1}$ sends the message to A_n and A_n sends the message to A_1(see Figure 3-2). In order to clarify this loop-free method, we assume that, in case of ties for the choice of neighbors at the current node, if one of the choices is the previous node, the *MFR* algorithm will select that node (that is, it will stop or flood the message). Therefore, such node is, by definition stated in Section 3.1, a *concave node* in this routing task using the *MFR* algorithm. According to the choice of neighbors and the *MFR* algorithm (using the dot product formulation given above) it follows that

$DA_n \cdot DA_1 > DA_2 \cdot DA_1$ since the node A_1 selects A_2, not A_n, to forward the message. Therefore $DA_n \cdot DA_1 > DA_1 \cdot DA_2 > DA_2 \cdot DA_3 > \ldots > DA_{n-1} \cdot DA_n > DA_n \cdot DA_1$, which is a contradiction.

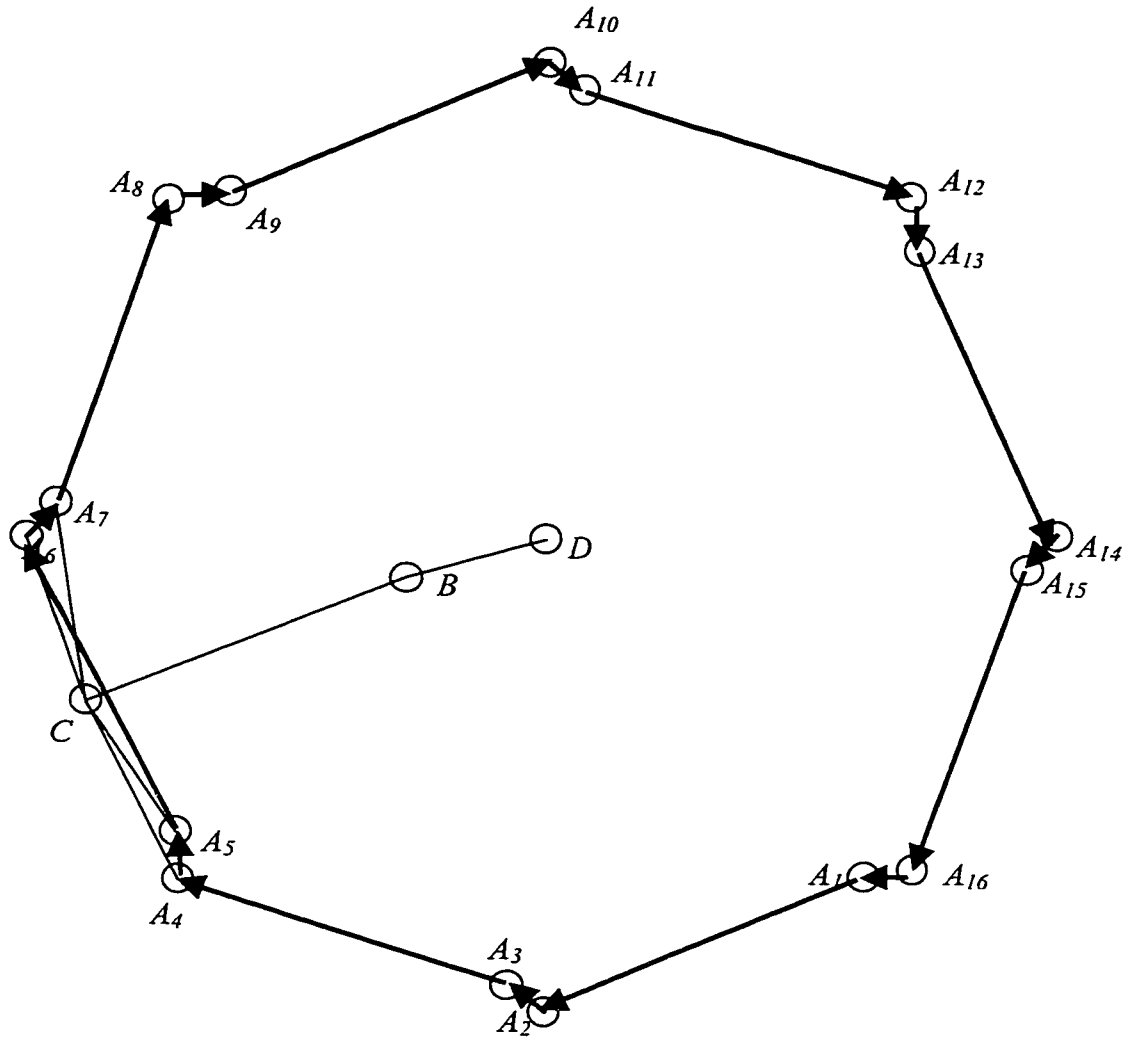


Figure 3-1. A loop in the directional routing

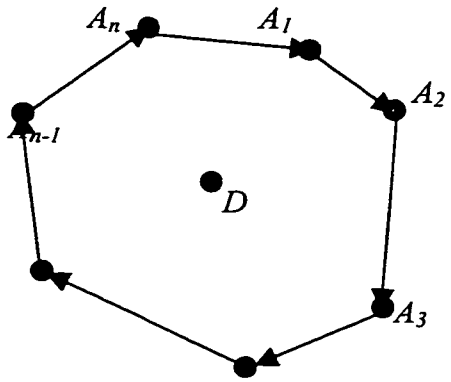


Figure 3-2. *MFR* and *GEDIR* algorithms are loop-free

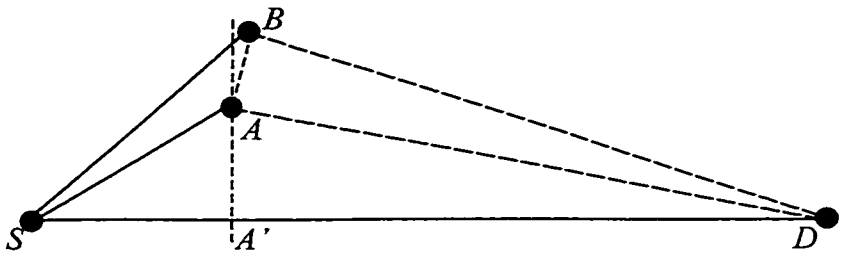


Figure 3-3. *GEDIR* and *MFR* may choose different node

3.2 GEographic DIstance Routing (GEDIR) Algorithm

We introduce a new routing algorithm for MANET, based on the locations (e.g. latitude and longitude) of all nodes. Each node is aware of geographic coordinates of all its direct neighbors. The sender node is also aware of the location of the destination, which is forwarded with the message. Node *A* that wants to send a message *m* to destination *D* uses the location information for *D* and for all its one hop neighbors to determine the neighbor *C* which is closest to *D* among all neighbors of *A*. The message is forwarded to *C*, and the same procedure is repeated until *D*, if possible, is eventually reached. Therefore, the algorithm is fully distributed. In example on Figure 2-2, sender *S* selects node *B* which is closer to *D* than *A*. The path selected by the algorithm is *SBEFGHID* and consists of seven hops.

It is noted that *GEDIR* algorithm is similar to that of *MFR* because both methods choose "forward" nodes. However, the difference lies in the "projection": *GEDIR* directly uses the Euclidean distance while *MFR* uses the projection of a neighbor onto the source-destination vector. Figure 3-3 shows a scenario where *GEDIR* and *MFR* pick different host for transmission: *GEDIR*

chooses node A for it is closer to D than B; while *MFR* selects node B since its projection is closer to D than that of A.

Note that, in this basic version, A does not compare its own distance against distances of its neighbors. Thus, even if A is closer to the destination than C, the message is still forwarded to C, with the hope that C will find another neighbor which is closer to destination than A is. Otherwise, C will return the message to A and a local loop (between A and C) is created. We will prove that this is the only kind of loop that may be formed in MANET using proposed distance based routing (unless nodes move so fast that from the moment the message is received to the time that a routing decision is made, the selected host is already outside the transmission range of current node). Since such loop can be easily detected by nodes A and C, they can stop forwarding m and prevent it from spinning between them. This simplest version of our algorithm will be referred to as GEographic DIstance Routing (*GEDIR*) algorithm.

3.3 Loop-freedom in GEDIR

The proof that *GEDIR* algorithm is inherently loop-free goes as follows. Again, we will prove it by contradiction. Suppose that there exists a loop in a distance routing algorithm, and let A_1 be the node on the loop that is closest to the destination (see Figure 3-2). Furthermore, we assume that this loop is not a local-loop. According to *GEDIR* algorithm A_1 forwards the message to its neighbor A_2, which then forwards to one of its neighbors, A_3 (following the created loop), which is closest to destination D among all neighbors of A_2. Following this logic, we then can establish the following: node A_{i+2} is closer to D than node A_i. So for $n = 2i$ (n is even number), we have A_n is closer to D than A_2. However, since A_1 forwards the message to its neighbor A_2 (in the loop), so A_2 is closer to D than A_n is, which is a contradiction. Similar proof for $n = 2i + 1$ (n is odd number).

This proof also suggests that, in case of equal distances from destination, current node should choose the node that forwarded the message to it. For instance, if $|A_1 A_2|=|A_2A_3|$, A_2 should send the message back to A_1, to avoid possible star shaped loop.

Assuming triangular inequality, both proofs of loop-free properties (for MFR and GEDIR algorithms) do not refer to the unit graphs and are valid in three-dimensional space. Thus, they are applicable to any model of MANET. The exclusion is, of course, the unrealistic case when nodes move purposely (combined with selected location update scheme) in such a way to maintain a loop (e.g. nodes of a regular polygon moving toward the center (destination) always just before the message is sent to them and returning back afterwards). However, since we are discussing routing algorithms in an ad hoc network, such scenario will only last a finite period of time by which at the end of this period the pattern will eventually be broken up, and so does the loop. Therefore, in the absence of such a purpose, message will exit such a temporary loop, and therefore we have shown the following:

Routing algorithms in wireless networks in which nodes forward the message to several neighbors closest to destination or with most forward progress (i.e. MFR and GEDIR algorithms and their enhancements: flooding, 2-hop, multiple path) are inherently loop-free.

From the proof above, we can further assert that if for a given routing task, there exists no concave node in the network, then the progress based routing algorithms will succeed. The proof is as follows:

Since there is no concave node in the network, thus no local-loop will be formed in the routing path. Furthermore, since we had already proven that the progressed based algorithms are loop-free,

so after a finite number (no more than $n - 1$, where n denotes total number of hosts in the network) of transmissions, the message will reach the destination (otherwise a loop would be formed, which contracts our assumption).

Since a node that has closer direction may be actually further away from destination, compass routing may exhibit loops, as shown in Figure 3-1. Note that the selected neighbor in *MFR* method may be also further from the destination, but the loop is never created. It is easy to find examples in which one of basic methods delivers the message to the destination while the others do not. Similarly, it is easy to construct examples in which the path length or number of hops for one method is smaller than for the other methods. Finally, one can construct examples showing that the ratio of hop count by one of algorithms over the shortest hop count may be arbitrarily large.

3.4 Variants of GPS based localized routing algorithms

It is obvious that these basic GPS based localized routing algorithms do not always achieve 100% delivery rate, and further enhancement is possible. Several variants are proposed in the following section aiming at improving the success rate.

3.4.1 2-hop variants

The delivery rate of *GEDIR*, *DIR* or *MFR* algorithms can be improved if nodes exchange information about their neighbors, and each node is aware of its 2-hop neighbors (neighbors of its neighbors). In this case, node A currently holding the message may choose the node closest to the destination D among all direct (1-hop) and 2-hop neighbors, and forward the message to its neighbor that is connected to the choice. In case of ties (that is, more than one neighbor connected to the closest 2-hop neighbor), choose the one that is closest to destination. We will refer to this

method as *2-hop GEDIR*. *2-hop DIR* and *2-hop MFR* can be similarly defined, by replacing all references to distance by direction and progress, respectively (with respect to *AD*). The abbreviated names *GEDIR-2, DIR-2* and *MFR-2* for 2-hop methods will be also used in the sequel. There are no multiple copies of the message in MANET at any transmission step in these 2-hop methods.

3.4.2 Flooding variants

We propose a modification to all three basic algorithms to avoid message dropping. Each algorithm proceeds as described until the message is supposed to be dropped by the corresponding algorithm at a concave node *A*. Modifications differ in the way concave nodes act. If an alternate network is available to the MANET for occasional use (for example, a satellite or other technology), the concave node may use it. Otherwise, we propose flooding as a solution. Full flooding, initiated at a concave node and performed afterwards at any node receiving the message, will certainly suffice to reach the destination, but the flooding rate will be affected. In order to enable this solution, messages should carry a bit of information about existence of a concave node on its previous path, so that receiving nodes may decide how to proceed. We propose to perform flooding only at concave nodes, while every other intermediate node should act with receiving message as in the corresponding basic routing algorithm. After forwarding the packet to all its neighbors, a concave node shall mark packet *id* in its entry corresponding to the given destination, and shall refuse to accept the same packet from any of its neighbors. Upon receiving a rejection message from a concave node, intermediate nodes will select the next best neighbor instead. In fact, the concave node has disconnected itself with respect to given packet. It is not necessary to carry additional flooding bit with the packet. The delivery of the packet to the destination is guaranteed (assuming that MANET is connected graph). The methods will be refereed to as

flooding GEDIR, *flooding DIR*, and *flooding MFR* routing methods (abbreviated as *f-GEDIR*, *f-DIR* and *f-MFR*). It is possible to construct examples showing that even full flooding at concave nodes does not guarantee message delivery unless concave nodes reject further copies of the same message.

3.4.3 Multiple path variants

Next, we propose *c-GEDIR*, *c-DIR* and *c-MFR* methods, in which message is initially sent to c neighbors which are closest to destination (whose direction or dot product are best, respectively), and afterwards, on intermediate nodes, it is forwarded to only one neighbor. These methods provide multiple paths (robustness) without much flooding. We shall describe three variants of *c-GEDIR* algorithm (by analogy, same three variants may apply to *c-DIR* and *c-MFR* methods). In the *original c-GEDIR* method, every intermediate node will forward the message to its best neighbor. Thus for $c=1$ it is equivalent to basic *GEDIR* method. Although the method may work without memorizing past traffic at each node, many nodes (close to destination) are anticipated to receive multiple copies of the message, and thus we implemented a variant in which every intermediate node will forward only the first received copy of each message. In the *alternate c-GEDIR* algorithm, each intermediate node forwards i-th received copy of the same message to i-th best (closest to destination) neighbor (for i=1, 2, 3, ...), disregarding neighbors from which the message came from. Thus, concave nodes do not stop transmitting in this method. In the *disjoint c-GEDIR* method, each intermediate node A, upon receiving the message, will forward it to its best neighbor among those who never received the same message before. After forwarding the message, node A becomes inactive with respect to that message, and rejects further copies of it. The *disjoint c-GEDIR* algorithm attempts to create c disjoint paths between source and destination

nodes. A node in *alternate* or *disjoint c-GEDIR* method stops forwarding the message if there is no enough neighbors to choose from for forwarding. Both methods are therefore loop-free although, in the *alternate c-GEDIR*, a message may return few times to the same node. Like *flooding* algorithms, each node in the multiple path algorithms remembers the recipient(s) of past messages.

The improvements mentioned in [BCSW, KV] for their directional algorithms to obtain actual protocols for each of our proposed algorithms can be easily incorporated, giving additional variety to geographically based routing methods. We note that flooding effect may be related to the urgency of the message itself; in other words, messages may have some priority identifiers that will be related to the flooding rate.

Chapter 4. New power-aware algorithms

In this chapter, we propose a new metric: the *power-cost* metric, some new power-aware algorithms [S] based on the existing and the new power metric. Before we proceed any further, let us look at some characteristics of power consumption in transmissions.

4.1 Some properties of power adjusted transmissions

In this section, we shall study the optimality of power-adjusted transmissions in MANETs, using a simple and general radio model. Three specific models (that is, metric for power consumption for sending and receiving message between two nodes at given distance) are considered in order to show the generality of some properties of power adjusted transmissions and generality of power aware routing algorithms to be described in the next section.

Two power-aware metrics [RM, HCB] are described in chapter 2. We shall here describe a third one, by modify the expression $P=T/d^{\alpha}$ as follows: $P=T/(1+d)^{\alpha}$. Thus, the received power is almost equal to the transmitted one when nodes are near each other (d near 0). Let d_0 be the distance that corresponds to the received power $P=T/2^{\alpha}$. That is, to $d=1$ in the expression. The value d_0 and the exponent α can be measured by physical experiments. The relative distance d is equal to the exact distance between nodes divided by d_0. In order to simplify our discussion, we shall assume $d_0=1$, thus d will be equated with the distance between nodes. Since all transmission powers are adjusted to minimum needed for successful transmission, P is a constant. Therefore the minimum power needed at the sender's node for successful transmission is $T=P(1+d)^{\alpha}$. We shall assume $\alpha=2$ in the sequel. The results for different values of α can be derived similarly. Nodes consume energy to receive packets. That energy is also constant (for constant packet length). We shall assume that the power needed for packet reception is eP, where e is constant that depends on

the type of equipment. For DEC Roambout radio [h] $e=1/2$, which is the value used in our experiments. To simplify further, we assume $P=1$ in the expressions that refer to power consumption. Thus, in this model, the power needed for transmission and reception at distance d is $u(d)=e+(1+d)^2$.

In order to include models that attenuate the power of signal of various exponents, we shall further generalize the model of [RM] and assume that the power needed for the transmission and reception of a signal is $u(d)=ad^{\alpha} + bd + c$. This model will encompass all three discussed models.

Suppose that the sender S is able to transmit the packet directly to the destination D. Let us examine whether energy can be saved by sending the packet to an intermediate node A between the two nodes and forwarding the packet from A to D (see Figure 4-1). Let $|SD|=d$, $|SA|=x$, $|AD|=d-x$.

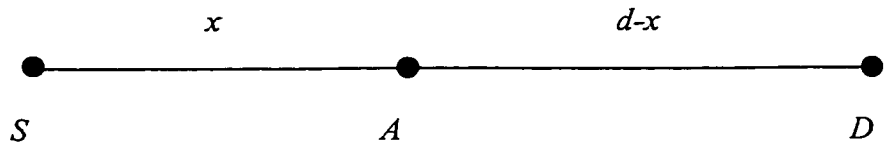


Figure 4-1. Saving energy by retransmission

Lemma 1. If $d>(c/(a(1-2^{1-\alpha})))^{1/\alpha}$ then there exists intermediate node A between source S and destination D so that the retransmission will save the energy. The greatest power saving is obtained when A is in the middle of SD.

Proof. The power needed to send message directly from S to D is $u(d)=ad^{\alpha} + bd + c$ while the power needed to send via A is $(ax^{\alpha} + bx + c) + (a(d-x)^{\alpha} + b(d-x) + c) = a(x^{\alpha} + (d-x)^{\alpha}) + bd + 2c$. $a(x^{\alpha} + (d-x)^{\alpha}) + bd + 2c < ad^{\alpha} + bd + c$ is satisfied for $g(x)=a(x^{\alpha} + (d-x)^{\alpha} - d^{\alpha}) + c < 0$. The minimum for $g(x)$ is obtained for $g'(x)=0$, i.e. $a(\alpha x^{\alpha-1} - \alpha(d-x)^{\alpha-1})=0$. Thus $x^{\alpha-1} = (d-x)^{\alpha-1}$, $x=d-x$, or $x=d/2$. The minimum is <0 if $g(d/2)<0$, i.e. $a((d/2)^{\alpha} + (d/2)^{\alpha} - d^{\alpha}) + c < 0$, or $ad^{\alpha}(2^{1-\alpha} -1)+c$

<0, which is satisfied for $c< ad^{\alpha}(1-2^{1-\alpha})$, or $d^{\alpha} > c/(a(1-2^{1-\alpha}))$, and lemma follows. Note that this inequality has a solution in d if and only if $\alpha>1$.

When $\alpha=2$, the following corollary follows for the two models that we adopted. We will give its full proof.

Corollary 1. If $d>(2c/a)^{1/2}$ then there exists intermediate node A between source S and destination D so that the retransmission will save the energy. The length of the interval of power saving intermediate nodes is $(d^2-2c/a)^{1/2}$. The greatest power saving is obtained when A is in the middle of SD. For the first model $c/a=e+1$, while for the second one $c/a=2E$.

Proof. The power needed to send message directly from S to D is $u(d)=ad^2 + bd +c$ while the power needed to send via A is $ax^2 + bx +c + a(d-x)^2 + b(d-x) +c = 2ax^2 -2adx + 2c + bd + ad^2$. $2ax^2 -2adx + 2c + bd + ad^2 < ad^2 + bd +c$ is satisfied for $2ax^2 - 2adx + c < 0$. The interval of power saving intermediate nodes is determined by solving the quadratic equation in x, and the lemma follows. Note that for $e=1/2$ in the first model one gets the threshold value of $d> 3^{1/2}=1.73$, while for $E=500$ in the second model we receive $d > 2(500^{1/2})=44.72$.

Lemma 2. If $d>(c/(a(1-2^{1-\alpha})))^{1/\alpha}$ then the greatest power savings are obtained when the interval SD is divided into $n>1$ equal subintervals, where n is the nearest integer to $d(a(\alpha-1)/c)^{1/\alpha}$. The minimal power is then $bd + dc(a(\alpha-1)/c)^{1/\alpha} + da(a(\alpha-1)/c)^{(1-\alpha)/\alpha}$.

Proof. Let SD be divided into intervals of lengths $x_1, x_2, \ldots, x_n$ such that $d=x_1+ \ldots + x_n$. The energy needed for transmissions using these intervals is $(ax_1^{\alpha}+bx_1+c) + \ldots + (ax_n^{\alpha}+bx_n+c) = nc + bd + a(x_1^{\alpha} + \ldots + x_n^{\alpha})$. For fixed x_i+x_j, the expression $x_i^{\alpha} + x_j^{\alpha}$ is minimal when $x_i=x_j$ (see the

proof of Lemma 1). Therefore the energy is minimal when $x_1=x_2= \ldots = x_n=d/n$ and is equal to $f(n)=cn + bd + an(d/n)^{\alpha} = nc + an^{1-\alpha}d^{\alpha} + bd$. This expression has the minimum when $f'(n)=0$, or when $c+a(1-\alpha)n^{-\alpha}d^{\alpha}=0$. i.e. $c=a(\alpha-1)n^{-\alpha}d^{\alpha}$, $n^{\alpha}=a(\alpha-1)d^{\alpha}/c$, $n=d(a(\alpha-1)/c)^{1/\alpha}$. Since n must be integer, one should round the value.

Corollary 2. Let $\alpha=2$. If $d > (2c/a)^{1/2}$ then the greatest power savings are obtained when the interval SD is divided into $n>1$ equal subintervals, where n is the nearest integer to $d(a/c)^{1/2}$.

Proof. Follows directly from Lemma 2. Let us verify whether power savings over direct transmission are obtained. For $n=d(a/c)^{1/2}$ the power consumption with n shorter transmissions is $2cd(a/c)^{1/2} + bd < ad^2 + bd + c$ since $2d(ac)^{1/2} < ad^2 + c$ follows from the inequality of arithmetic and geometric means. Thus, power savings are indeed obtained.

Assuming that we can set additional nodes in arbitrary positions between the source and destination, the following theorem gives optimal power saving in packet transmissions.

Theorem 1. Let d be the distance between the source and the destination. The power needed for direct transmission is $u(d)=ad^{\alpha} + bd + c$ which is optimal if $d \le (c/(a(1-2^{1-\alpha})))^{1/\alpha}$. Otherwise (that is, when $d > (c/(a(1-2^{1-\alpha})))^{1/\alpha}$), $n-1$ equally spaced nodes can be selected for retransmissions, where $n= d(a(\alpha-1)/c)^{1/\alpha}$ (rounded to nearest integer), producing minimal power consumption of about $v(d)= bd + dc(a(\alpha-1)/c)^{1/\alpha} + da(a(\alpha-1)/c)^{(1-\alpha)/\alpha}$. For $\alpha=2$, the power needed for direct transmission is $u(d)=ad^2 + bd + c$ which is optimal if $d \le (2c/a)^{1/2}$. Otherwise (that is, when $d > (2c/a)^{1/2}$), $n-1$ equally spaced nodes can be selected for retransmissions, where $n= d(a/c)^{1/2}$ (rounded to nearest integer), producing minimal power consumption of about $v(d)=2d(ac)^{1/2}+ bd$.

Theorem 1 announces the possibility of converting polynomial function in d (with exponent α) for power consumption (in case of direct transmission from sender to destination) to linear function in d by re-transmitting the packet via some intermediate nodes that may be available in MANET.

4.2 Power-cost metrics

In this section, we shall present a new metric: the *power-cost* function. As the name suggests, this function combines two already know power metrics: *transmission cost* and *power consumption*. Assuming that currently node B is holding the message that is intended to D, and node A, distance r, is a neighbor node of B (in Figure 4-2). Then the power-cost for B to forward the message to A is $power\text{-}cost(A, r) = u(r) \cdot f(A)$ where $u(r)$ denotes the power required for transmission to A and $f(A)$ represents the transmission cost should A receive the message. Implemented in Dijkstra's shortest weighted path algorithm where each edge is assigned with $power\text{-}cost(A, r) = (ad^{\alpha} + bd + c) \cdot f(A)$, and we have the *SP-Power-Cost* algorithm which would minimize the total power consumption for the routing while extending as much as possible the lifetime of each node.

A variant of the *SP-Power-Cost* algorithm that uses different power-cost function is also studied. Here is the new power-cost function: $power\text{-}cost1(A, B, r) = u(r) \cdot f'(B) + u(r') \cdot f(A)$. Note that the function $f'(B)$ and r' in $u(r)$ are different from their counter part $f(A)$ and r in $u(r)$.

$f(A)$ is the inverse of the normalized power level $g(A)$ at node A. $f'(B)$ is defined as the following: $f'(B) = \text{avg } (f(X))$ where X are the neighbor nodes of B. In other word, $f'(B)$ is the average of *transmission cost* of all the neighbor nodes of B, thus $f'(B)$ is fixed for all neighbors of the current host B.

Similar to the definition of $f'(B)$, r' is defined as the average length of the edges connecting the current node (node B in Figure 4-2) to all its neighbors. In Figure 4-2 the r' is equal to r since A is the only neighbor of B, but in Figure 4-3 the $r' = (r_1 + r_2) / 2$ for node A_1.

Hence, the new *SP-Power-Cost* algorithm that assigns $power\text{-}cost1(A, B, r) = u(r) \bullet f'(B) + u(r') \bullet f(A)$ to each edge will be referred as *SP-Power-Cost1* algorithm, and we shall examine both algorithms in our experiments.

4.3 Power-aware routing algorithms

If nodes have information about position and activity of all other nodes in the network (or if the decisions are made centrally), then the optimal power saving algorithm that will minimize the total energy per packet, can be obtained by applying Dijkstra's single source shortest weighted path algorithm, where each edge has weight $u(d)=ad^{\alpha} + bd + c$, where d is the length of the edge (that is, the relative distance between the two nodes). This will be referred as the *SP-power algorithm.*

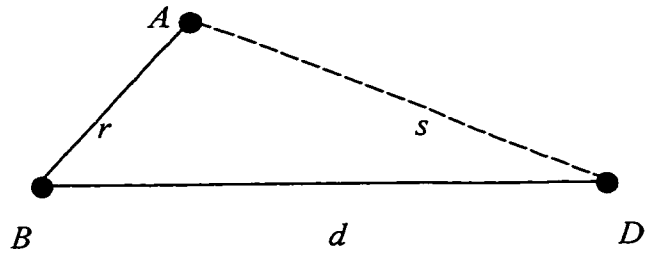


Figure 4-2. Distributed power-conserving routing algorithm

4.3.1 Power efficient routing algorithms

We shall now describe a corresponding power efficient distributed routing algorithm [S]. In a distributed algorithm, each node makes its routing decision based on local information, without assistance of any central facility. Each node is assumed to know the location of itself, all its

neighboring nodes, and the destination node, and is aware of inactive neighboring nodes. The existence of inactive neighbors prohibits the use of shortest weighted path routing algorithms unless the activity information for each node is broadcast to other nodes. The source (or any intermediate node) S should select one of its neighbors to forward packet toward destination, with the goal of reducing the total power needed for the packet transmission. Let A be a neighbor of B, and let $r=|AB|$, $d=|BD|$, $s=|AD|$ (see Figure 4-2).

The power needed for transmission from B to A is $u(r)=ar^{\alpha} + br + c$, while the power needed for the rest of routing algorithm is not known. Assuming uniformly distributed network, we shall make a fair assumption that the power consumption for the rest of routing algorithm is equal to the optimal one, as outlined in Theorem 1. That is, the power needed for transmitting message from A to D is estimated to be $v(s)= bs + sc(a(\alpha-1)/c)^{1/\alpha} + sa(a(\alpha-1)/c)^{(1-\alpha)/\alpha}$. For $\alpha=2$, $v(s)=2s(ac)^{1/2} + bs$. This is, of course, an unrealistic assumption. However, it is fair to all nodes. A more realistic assumption might be to multiply the optimal power consumption by a factor t, which is a constant that depends on the network, and is equal to the average power consumption per packet (obtained experimentally) divided by the optimal one.

The distributed *power efficient routing algorithm* can be described as follows. Each node B (source or intermediate node) will select one of its neighbors A which will minimize $p(B,A)=u(r)+v(s)= ar^{\alpha} + br + c + bs + sc(a(\alpha-1)/c)^{1/\alpha} + sa(a(\alpha-1)/c)^{(1-\alpha)/\alpha}$. For $\alpha=2$ it becomes $u(r)+v(s)= ar^2 + br + c + 2s(ac)^{1/2} + bs$. If destination D is a neighbor of B then compare the expression with the corresponding one, $u(d)=ad^2 + bd + c$, needed for direct transmission. Deliver the packet directly to D if it reduces the energy. In fact, $s=0$ for D, and D can be treated as any other neighbor. The algorithm proceeds until the destination is reached, if possible. A generalized

power efficient routing algorithm may attempt to minimize $p(B,A)=u(r)+tv(s)$, where t is a network parameter (however, we only experimented with $t=1$).

In the basic (experimental) version of the algorithm (and in the remaining distributed algorithms presented below), the transmission stops if message is to be returned to a neighbor it came from (otherwise, a detectable loop is created). Various flooding based or multiple path techniques [SL1] can be added to the protocol if delivery rate is to be improved; however, their total power consumption needs to be studied. The power-efficient routing algorithm may be formalized as follows.

Power-routing(S,D);
A:=S;
Repeat
B:=A;
Let *A* be neighbor of *B* that minimizes $p(B,A)=u(r)+ tv(s)$;
Send message to *A*
until *A=D* (* destination reached *) or *A=B (* delivery failed *);*

4.3.2 Cost efficient routing algorithms

Let us now consider the second metric proposed in [SWR], measuring the nodes lifetime. Recall that the cost of each node is equal to $f(A)=1/g(A)$ where $g(A)$ denotes the remaining lifetime, normalized to be in the interval [0,1]. [SWR] proposed shortest weighted path algorithm based on this node cost. The distributed version of this algorithm, assuming constant power for each transmission, can be designed as follows.

The cost $c(A)$ of a route from *B* to *D* via neighboring node *A* is the sum of the cost $f(A) =1/g(A)$ of node *A* and the estimated cost of route from *A* to *D*. The cost $f(A)$ of each neighbor *A* of node *B* currently holding the packet is known to *B*. What is the cost of other nodes on the remaining path?

We assume that this cost is proportional to the number of hops between A and D. The number of hops, in turn, is proportional to the distance $s=|AD|$, and inversely proportional to radius R. Thus, the cost is ts/R, where factor t is to be investigated separately. Its best choice might even be determined by experiments. We have considered the following choices for factor t:

i) t is a constant number, which may depend on network conditions,
ii) $t=f(A)$ (that is, assuming that remaining nodes have equal cost as A itself),
iii) $t=f'(A)$, where $f'(A)$ is the average value of $f(X)$ for A and all neighbors X of A,
iv) $t=1/g'(A)$, where $g'(A)$ is the average value of $g(X)$ for A and all neighbors X of A,
v) $t=f'(B)$ (that is, assuming that remaining nodes have equal cost to the cost of B),
vi) $t=1/g'(B)$.

Note that $t=t(A)$ in i-iv is a function of A only, which is not the case for v-vi. Thus the cost $c(A)$ of a route from S to D via neighboring node A is estimated to be $c(A)=f(A)+ts/R$, for the appropriate choice of t. Although this seems to be the natural choice, it is not clear whether such definition of $c(A)$ will give the best experimental results. We therefore suggest to investigate also the product of two contributing elements instead of their sum, that is the cost definition $c(A)= f(A)ts/R$. The corresponding *SP-Cost* algorithm will then use corresponding *cost* as the weight of each edge.

The distributed *cost efficient routing algorithm* [S] can be described as follows. If destination is one of neighbors of node B currently holding the packet then the packet will be delivered to D. Otherwise, B will select one of its neighbors A which will minimize $c(A)$. The algorithm proceeds until the destination is reached, if possible, or until a node selects the neighbor the message came from as its best option to forward the message (i.e. message is delivered to a concave node). The algorithm can be coded as follows:

Cost-routing(S,D);

A:=S;

Repeat

B:=A;

Let *A* be neighbor of *B* that minimizes *c(A)*;

If *D* is neighbor of *B*

then send to *D* **else** send to *A*

until *D* is reached or *A=B;*

4.3.3 Power-cost efficient routing algorithms

We may incorporate both power and cost considerations into a single routing algorithm. A new power-cost metrics is first introduced here. What is the power-cost of sending a message from node *B* to node the neighboring node *A*? We propose two different ways to combine power and cost metrics into a single power-cost metric, based on the product and sum of two metrics, respectively. If the product is used, then the power-cost of sending message from *B* to a neighbor *A* is equal to *power-cost(B,A)=f(A)u(r)* (where $|AB|=r$). *The SP-power-cost algorithm* can find the optimal power-cost by applying Dijkstra's single source shortest weighted path algorithm (the node cost is transferred to the edge leading to the node). The sum, on the other hand, leads to a new metrics *power-cost(B,A)=αu(r)* + *βf(A)*, for suitably selected values of α and β. For example, sender node *S* may fix $\alpha=f'(S)$ and $\beta=u(r')$, where r' is the average length of all edges going out of *S*. The values α and β are (in this version) determined by *S* and used, without change, by other nodes *B* on the same route. The corresponding SP-power-cost algorithm will also use such defined metric.

The *power-cost efficient routing algorithm* [S] may be described as follows. Let *A* be the neighbor of *B* (node currently holding the message) that minimizes *pc(B,A)= power-cost(B,A)* +

v(s)f'(A) (where *s=0* for *D*, if *D* is a neighbor of *B*). The packet is delivered to *A*. Thus, the packet is not necessarily delivered to *D*, when *D* is a neighbor of *B*. The algorithm proceeds until the destination is reached, if possible. The algorithm may be coded as follows.

Power-cost-routing(S,D);

A:=S;

Repeat

B:=A;

Let *A* be neighbor of *B* that minimizes *pc(B,A)= power-cost(B,A) + v(s)f'(A);*

Send message to *A*

until *A=D* (* destination reached *) or *A=B (* delivery failed *);*

The algorithm may be improved in several ways. The second term may be multiplied by a factor that depends on network conditions. If the necessary information about 2-hop neighbors is available, all distributed algorithms may use it to make better decision, similar to 2-hop *GEDIR*, *DIR*, and *MFR* algorithms.

A variant of the *power-cost efficient routing algorithm* is also suggested with the goal to improve the energy consumption. The variant differs at the point when destination node *D* is within the reach of the intermediate node. When the node (say node *A*) holding the message *m* sees that the destination node *D* is within its transmission radius, it then applies the *power-efficient routing algorithm* instead of the *power-cost efficient routing algorithm*. The cause of this modification is that sometimes when the destination node *D* has lower battery life then its neighbors, the original algorithm tends to transmit the message around *D*'s neighbors until their power levels had dropped roughly close to that of *D's* before finally delivers the message *m* to *D*. Since node *D* will eventually receives the message, thus it makes no sense to add the extra hops which leads to additional energy consumption uselessly. Thus switching to *power-efficient routing*

algorithm at the last step will decrease the total power consumption. Here is the modified algorithm (called *Power-cost-routing1*):

Power-cost-routing1(S,D);

A:=S;

Repeat

B:=A;

If *D* is within transmission radius of *B* then

Let *A* be neighbor of *B* that minimizes $p(B,A)=u(r)+tv(s)$;

Else

Let *A* be neighbor of *B* that minimizes $pc(B,A)=power\text{-}cost(B,A)+v(s)f'(A)$;

End-If

Send message to *A*

until *A=D* (* destination reached *) or *A=B (* delivery failed *);*

4.4 Loop-freedom in power-aware routing algorithms

Theorem 2. The localized power efficient routing algorithm is loop-free.

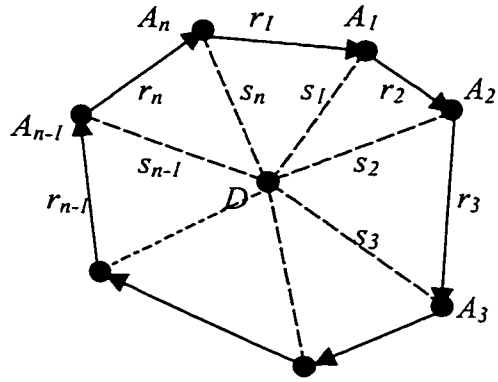


Figure 4-3. Power efficient routing algorithm is loop-free

Proof. Suppose that, on the contrary, there exists a loop in the algorithm. Let A_1, A_2, ... A_n be the nodes in the loop, so that A_1 send the message to A_2, A_2 sends the message to A_3, ..., A_{n-1} sends the message to A_n and A_n sends the message to A_1 (see Figure 4-3). Let s_1, s_2, ..., s_n be the distances of

A_1, A_2, ... A_n from D, respectively, and let $|A_nA_1|=r_1$, $|A_1A_2|=r_2$, $|A_2A_3|=r_3$, ..., $|A_{n-1}A_n|=r_n$. Let $u(r)= ar^{\alpha} + br + c$ and $v(s) = bs + sc(a(\alpha-1)/c)^{1/\alpha} + sa(a(\alpha-1)/c)^{(1-\alpha)/\alpha}$ (for $\alpha=2$, $v(s)= 2s(ac)^{1/2} + bs$). According to the choice of neighbors in Figure 4-3 it follows that $u(r_1)+v(s_1)<u(r_n)+v(s_{n-1})$ since the node A_n selects A_1, not A_{n-1}, to forward the message. Similarly $u(r_2)+v(s_2) < u(r_1)+v(s_n)$ since A_1 selects A_2 rather than A_n. Next, $u(r_3)+v(s_3) < u(r_2)+v(s_1)$, ..., $u(r_n)+v(s_n) < u(r_{n-1})+v(s_{n-2})$. By summing left and right sides we obtain $u(r_1)+u(r_2)+...+u(r_n)+v(s_1)+v(s_2)+...+v(s_n) < u(r_n)+u(r_1)+...+u(r_{n-1})+v(s_{n-1})+v(s_n)+...+v(s_{n-2})$ which is a contradiction since both sides contain the same elements. Thus, the algorithm is loop-free.

In order to provide loop-freedom for the method, we assume that (for this and other mentioned methods below), in case of ties for the choice of neighbors, if one of choices is the previous node, the algorithm will select that node (that is, it will stop or flood the message). Note that the above proof may be applied (by replacing '+' with '*') to an algorithm that will minimize $p(A)=u(r)tv(s)$.

Theorem 3. Localized cost efficient algorithms are loop-free, for parameter values t given by i-iv.

Proof. Note that the cost $c(A)$ of sending message from B to A in cases i-iv is only the function of A (that is, $t=t(A)$), and is independent on B (this is not the case with options v-vi). The proof is by contradiction, similar to the proof of previous theorem. Suppose that there exists a loop in the algorithm. Let A_1, A_2, ... A_n be the nodes in the loop (see Figure 4-3). Let $c(A_1)$, $c(A_2)$, ... $c(A_n)$, be the costs of sending message to nodes A_1, A_2, ... A_n, respectively, from the previous node in the loop. According to the choice of neighbors in Figure 4-3 it follows that $c(A_1) < c(A_{n-1})$ since the node A_n selects A_1, not A_{n-1}, to forward the message. Similarly $c(A_2) < c(A_n)$ since A_1 selects A_2 rather than A_n. Next, $c(A_3) < c(A_1)$, ..., $c(A_n) < c(A_{n-2})$. By summing left and right sides we obtain $c(A_1)+c(A_2)+...+c(A_n) < c(A_{n-1})+c(A_n)+...+c(A_{n-2})$ which is a contradiction since both sides

contain the same elements. Thus, the algorithm is loop-free. The proof is valid for both formulas $c(A)=f(A)+ts/R$ and $c(A)=f(A)ts/R$. Note that the proof assumes that the cost of each node is not updated (that is, communicated to the neighbors) while the routing algorithm is in progress. It is possible to show that, on the other hand, if nodes inform their neighbors about new cost after every transmitted message, a loop (e.g. triangle) can be formed.

Theorem 4. Localized power-cost efficient algorithms are loop-free for the metrics *power-cost(B,A)*$=\alpha u(r) + \beta f(A)$ (where α and β are arbitrary constants), and $pc(B,A)=$ *power-cost(B,A)* + $v(s)t(A)$ (where $t(A)$ is determined by one of formulas i-iv).

Proof. The proof is again by contradiction, similar to the proof of previous theorems. Suppose that there exists a loop $A_1, A_2, \ldots A_n$ in the algorithm (see Figure 4-3). Let $pc(A_n, A_1), pc(A_1, A_2), \ldots pc(A_{n-1}, A_n)$, be the power-costs of sending message to nodes $A_1, A_2, \ldots A_n$, respectively, from the previous node in the loop. According to the choice of neighbors in Figure 4-3 it follows that $pc(A_n, A_1) < pc(A_n, A_{n-1})$ since the node A_n selects A_1, not A_{n-1}, to forward the message. Similarly $pc(A_1, A_2) < pc(A_1, A_n)$, $pc(A_2, A_3) < pc(A_2, A_1)$, …, $pc(A_{n-1}, A_n) < pc(A_{n-1}, A_{n-2})$. By summing left and right sides we obtain $pc(A_n, A_1) + pc(A_1, A_2) + pc(A_2, A_3) + \ldots + pc(A_{n-1}, A_n) < pc(A_n, A_{n-1}) + pc(A_1, A_n) + \ldots + pc(A_{n-1}, A_{n-2})$. This inequality is equivalent to $[\alpha u(r_n) + \beta f(A_1) + v(s_1)t(A_1)] + [\alpha u(r_1) + \beta f(A_2) + v(s_2)t(A_2)] + \ldots + [\alpha u(r_{n-1}) + \beta f(A_n) + v(s_n)t(A_n)] < [\alpha u(r_n) + \beta f(A_{n-1}) + v(s_{n-1})t(A_{n-1})] + [\alpha u(r_1) + \beta f(A_n) + v(s_n)t(A_n)] + \ldots + [\alpha u(r_{n-1}) + \beta f(A_{n-2}) + v(s_{n-2})t(A_{n-2})]$, which is a contradiction since both sides contain the same elements. Thus, the algorithm is loop-free. Note that the proof also assumes that the cost of each node is not updated (that is, communicated to the neighbors) while the routing algorithm is in progress. Note that this proof does not work for the

formula *power-cost(B,A)=f(A)u(r),* which does not mean that the corresponding power-cost routing algorithm is not loop-free.

Corollary 3. Localized power-cost1 efficient algorithms is loop-free for the metrics *power-cost(B,A)=αu(r)* + *βf(A)* (where α and β are arbitrary constants), and *pc(B,A)= power-cost(B,A)* + *v(s)t(A)* (where *t(A)* is determined by one of formulas i-iv).

Proof. The proof is again by contradiction, similar to the proof of previous theorems. Suppose that there exists a loop $A_1, A_2, \ldots A_n$ in the algorithm (see Figure 4-3). Then we can assert that the destination node D is NOT within transmission radius of any of the node A_i's. This can be proved by the **Theorem 2**. Then follow the proof of **Theorem 4**: Let $pc(A_n, A_1), pc(A_1, A_2), \ldots pc(A_{n-1}, A_n)$, be the power-costs of sending message to nodes $A_1, A_2, \ldots A_n$, respectively, from the previous node in the loop. According to the choice of neighbors in Figure 4-3 it follows that $pc(A_n, A_1) < pc(A_n, A_{n-1})$ since the node A_n selects A_1, not A_{n-1}, to forward the message. Similarly $pc(A_1, A_2) < pc(A_1, A_n)$, $pc(A_2, A_3) < pc(A_2, A_1)$, …, $pc(A_{n-1}, A_n) < pc(A_{n-1}, A_{n-2})$. By summing left and right sides we obtain $pc(A_n, A_1) + pc(A_1, A_2) + pc(A_2, A_3) + \ldots + pc(A_{n-1}, A_n) < pc(A_n, A_{n-1}) + pc(A_1, A_n) + \ldots + pc(A_{n-1}, A_{n-2})$. This inequality is equivalent to $[\alpha u(r_n) + \beta f(A_1) + v(s_1)t(A_1)] + [\alpha u(r_1) + \beta f(A_2) + v(s_2)t(A_2)] + \ldots + [\alpha u(r_{n-1}) + \beta f(A_n) + v(s_n)t(A_n)] < [\alpha u(r_n) + \beta f(A_{n-1}) + v(s_{n-1})t(A_{n-1})] + [\alpha u(r_1) + \beta f(A_n) + v(s_n)t(A_n)] + \ldots + [\alpha u(r_{n-1}) + \beta f(A_{n-2}) + v(s_{n-2})t(A_{n-2})]$, which is a contradiction since both sides contain the same elements. Thus, the algorithm is loop-free. Note that the proof also assumes that the cost of each node is not updated (that is, communicated to the neighbors) while the routing algorithm is in progress. Note that this proof does not work for the formula *power-cost(B,A)=f(A)u(r),* which does not mean that the corresponding power-cost routing algorithm is not loop-free.

Chapter 5. Performance evaluation of GEDIR and related constant-metric algorithms

In this chapter we would discuss simulation experiment run to study the performance of *GEDIR* compared to other known GPS based constant-metric algorithms and their corresponding variants.

5.1 Experiment parameters

This experiment evaluates the performance of basic, *2-hop*, *flooding* and *multiple path GEDIR*, *DIR* and *MFR* methods. For each selected pair *(n,d)*, 20 connected graphs are randomly generated. In each graph, each of n nodes is chosen by selecting its x and y coordinates at random in the interval *[0,100)*. In order to control the average node degree d, we sort all $n(n-1)/2$ (potential) edges in the network by their length, in increasing order. The radius R that corresponds to chosen value of d is equal to the length of $nd/2$-th edge in the sorted order. Generated graphs that were disconnected are ignored, and additional ones are regenerated. We experimented with the following network sizes: n= 20, 50, and 100. For n= 20, the average degrees tested were d= 2, 3, 4 and 5; for n= 50, d ranges between 4 and 8; and for n= 100, d is between 4 and 14. For each graph, 100 random source-destination pairs are chosen, and the routing was performed in those pairs. Averages over all 20 graphs with the same parameters are then found. We shall present here only some of results for $n=100$.

LAR2 scheme from [KV] is added in the experiments, since it had best performance among schemes proposed by the same authors, according to their measurements. In one transmission step (of broadcast type), the source or each intermediate node A will forward the message to all nodes that are closer to the destination than A is (thus we selected value $\delta=0$). Authors did not mention whether nodes memorize messages to reduce flooding rate. Experiments in [CL] compared ants

based method with *LAR2* without memorizing past traffic and reported flooding ratio in *LAR2* over thousand times higher than in ant based method. We therefore assumed that nodes in *LAR2* do memorize messages and do not transmit the same message more than once. Nodes in *LAR2* that have no closer neighbor to destination than themselves do not retransmit the message. In case of no recipients found, ALL neighbors that had not received the message will be transmitted. Thus, the flooding rate in *LAR2* is simply the ratio of nodes that transmit the message. Possible message collisions in *LAR2,* flooding and multiple path methods are ignored in our experiments.

Since some of the algorithms tested here do not have the loop-free property, so additional checking mechanism was added in the program to stop the simulation (and mark the result as failure) when the total hop count (accumulated) reaches the maximum. The node number n is used as the threshold since we are assuming that no loop would be formed, so each node will retransmit the message at most once, thus at most n hops.

5.2 New analytical tool

To better analyze each algorithm individually, another software package was developed to display graphically the routing (successful or non-successful) path of each algorithm at each test trial. With this program, we were able to do some more in-depth analysis to provide better and more accurate explanation on certain behavior of the algorithms. The following images demonstrate the usage of this tracing software:

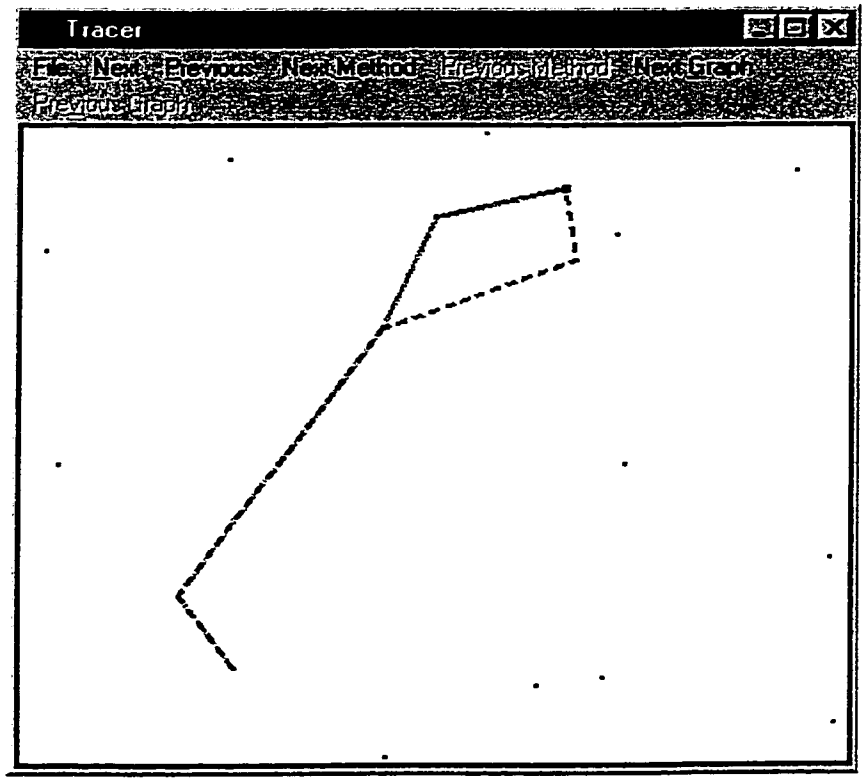


Image 5-1. *DIR* routing vs. *SP* routing at n=20

From Image 5-1, the black dots represent each individual host in the network, the slightly larger one representing the destination node. The black dotted line represents the path used by Shortest Path algorithm. The gray line is the routing path under the DIR protocol which also successfully delivery the message. Notice that the starting and finishing point of both paths coincide with each other. In case where the routing method has failed, the path would only be shown from the starting point until to the node where the message was dropped, and the color is changed to dark gray. In this example, the DIR is using same number of hops as SP, although their paths are different.

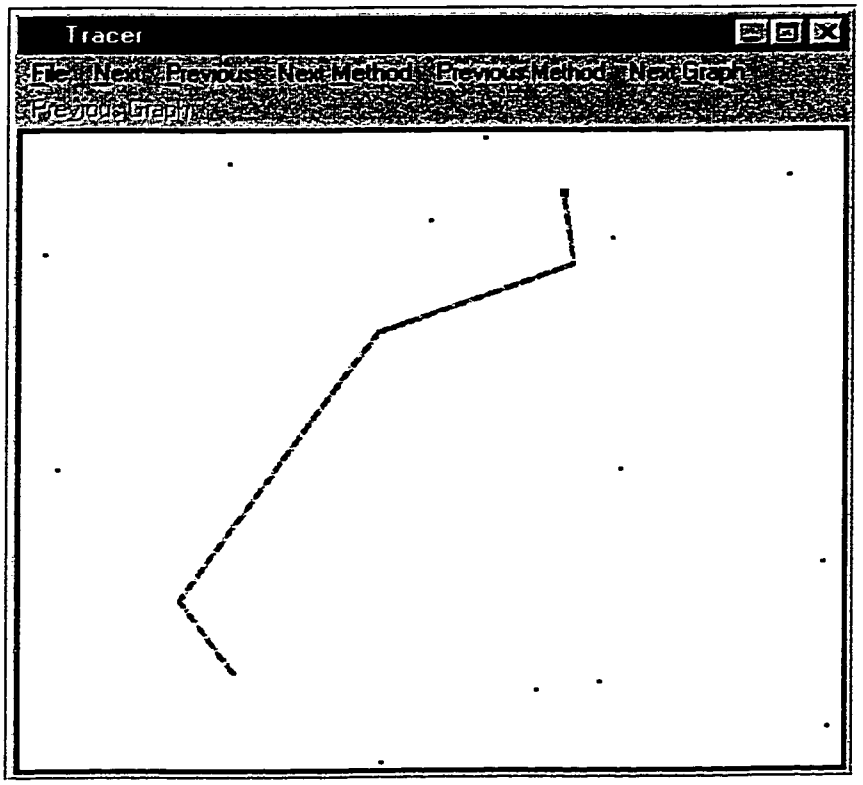


Image 5-2. *GEDIR* routing vs. *SP* routing

The Image 5-2 displays the GEDIR routing for the same starting and ending point. Notice that there is only one visible path on the graph. The reason is that the path found by GEDIR is the same as the SP, thus both paths overlap each other on the graph (dotted line overlapped by gray line). In this particular case, the GEDIR algorithm had found the same path as the SP, but the path discovered by GEDIR is different from the path discovered by DIR in Image 5-1 (different choice of neighbor at the before-last transmission).

5.3 Performance evaluation

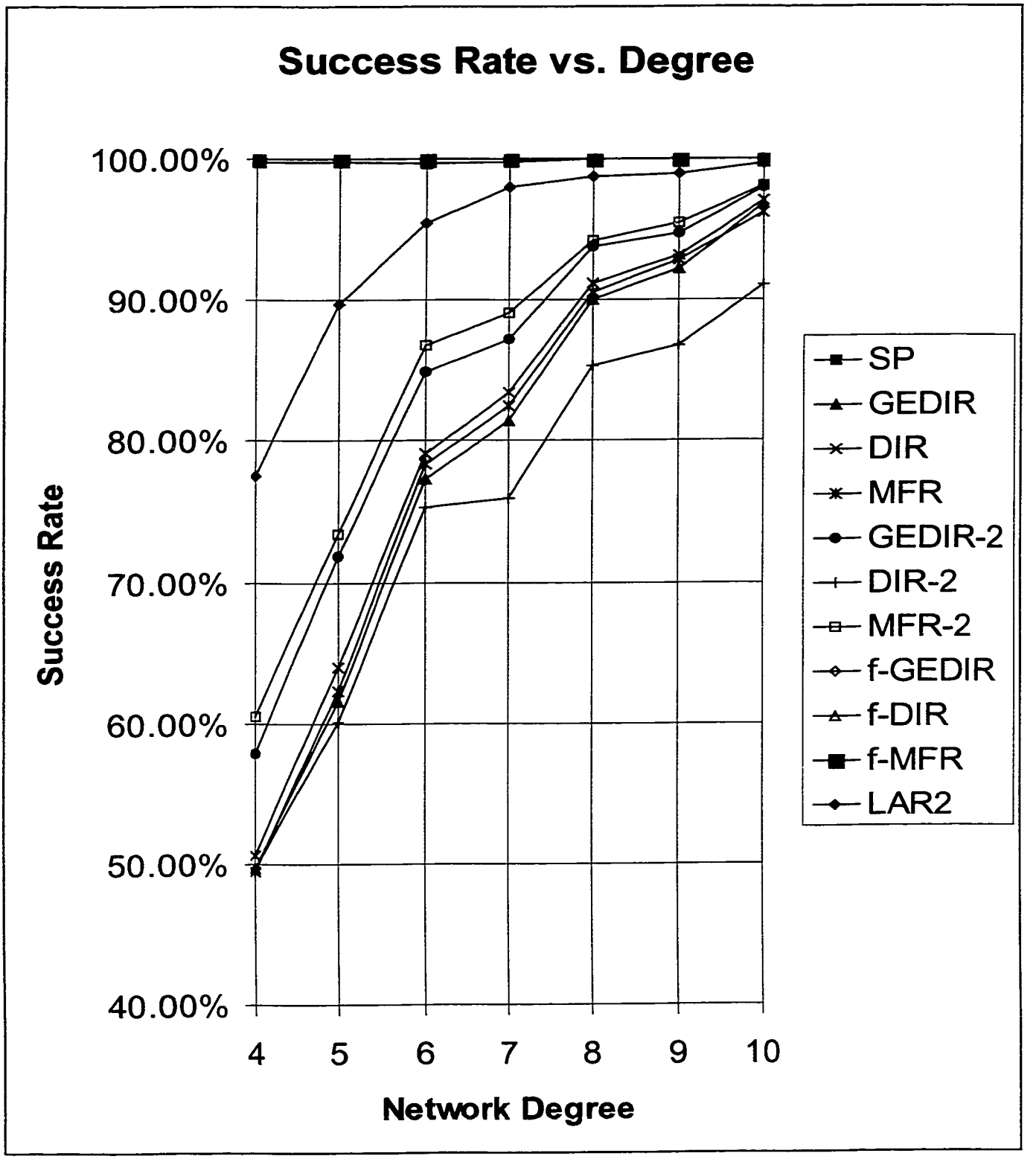
Because of large amount of data collected for different network size n, only simulation results for n=100 nodes are discussed here. However same analysis can be applied to different network size of n=20 and 50 nodes.

5.3.1 Success rate for individual methods

Table 5-1 (and Graph 5-1) show the delivery rates for n=100 nodes. The success rates for *DIR*, *GEDIR* and *MFR* methods are comparable (about 50% for d=4, 62-64% for d=5, 77-79% for d=6, 81-83% for d=7, about 90%, 93% and 97% for d=8, 9, 10, respectively). Thus, success rate greatly depends on network degree but not much on basic method selected! While the success rate for the very basic method on high degree network is already impressive (over 90%), very low degree networks require enhancements to basic methods (e.g. half messages not delivered for d=4). *GEDIR-2* (2-hop *GEDIR)* and *MFR-2* have increased their success rates compared to 1-hop variants (by 7-10% for low degrees, 1% for high degrees) while *2-DIR* decreased its success rate for 1-8% compared to *DIR*. The reason for the drop of success rate for *2-DIR* method is that a 2-hop neighbor C of A with closest direction AC with respect to AD may be very far from optimal direction with respect to BD where B is the common neighbor of A and C. *f-GEDIR* and *f-MFR* have *100%* success as expected, while *f-DIR* may fail (due to possible undetected loop creation). *LAR2* method did not offer reliable success at low degrees (78% for d=4) and was inferior to flooding methods.

Degree	4	5	6	7	8	9	10
SP	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
GEDIR	49.70%	61.55%	77.30%	81.40%	90.05%	92.25%	96.80%
DIR	50.60%	63.95%	79.10%	83.35%	91.20%	93.20%	97.05%
MFR	49.50%	62.30%	78.40%	82.45%	90.50%	92.85%	96.20%
GEDIR-2	57.90%	71.85%	84.90%	87.15%	93.75%	94.75%	98.05%
DIR-2	49.70%	60.05%	75.30%	75.90%	85.25%	86.75%	91.10%
MFR-2	60.45%	73.45%	86.80%	89.10%	94.25%	95.50%	98.15%
f-GEDIR	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
f-DIR	99.75%	99.75%	99.70%	99.80%	99.95%	100.00%	100.00%
f-MFR	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
LAR2	77.50%	89.75%	95.45%	98.05%	98.75%	99.00%	99.65%

Table 5-1. Delivery rates for n=100 nodes



Graph 5-1. Delivery rates for n=100 nodes

Degree	4	5	6	7	8	9	10
SP	0	0	0	0	0	0	0
GEDIR	0.503	0.3845	0.227	0.186	0.0995	0.0775	0.032
DIR	0.493	0.3605	0.2085	0.1665	0.088	0.068	0.0295
MFR	0.505	0.377	0.216	0.1755	0.095	0.0715	0.038
GEDIR-2	0.421	0.2815	0.151	0.1285	0.0625	0.0525	0.0195
DIR-2	0.316	0.287	0.19	0.202	0.132	0.1185	0.081
MFR-2	0.383	0.2555	0.128	0.106	0.054	0.0445	0.017
f-GEDIR	11.055	7.2915	3.3475	2.54	0.9325	0.6505	0.229
f-DIR	10.568	6.575	3.0915	2.0615	0.8405	0.6505	0.2305
f-MFR	10.8445	7.4745	3.1905	2.47	0.907	0.6425	0.294
LAR2	6.3565	8.6435	11.41	12.771	13.4945	15.584	16.178

Table 5-2. Concave Node Ratio for n=100 nodes

If we look at the number on Table 5-2 (concave node ratio) and Table 5-1 (success rate) for the basic methods, there is a very interesting fact: the sum of concave node ratio and the success rate for the basic methods is always 1. This means that the in all the failed cases for basic methods, they all resulted from transmission dropped at a concave node.

The higher number of concave nodes that was found in flooding methods tells us that several flooding was occurred during the transmission and several different paths were created. Among those paths usually only one reaches the destination, thus all other routes ended up in some nodes whose neighbors had already been flooded, or all its neighbors had become 'dormant'.

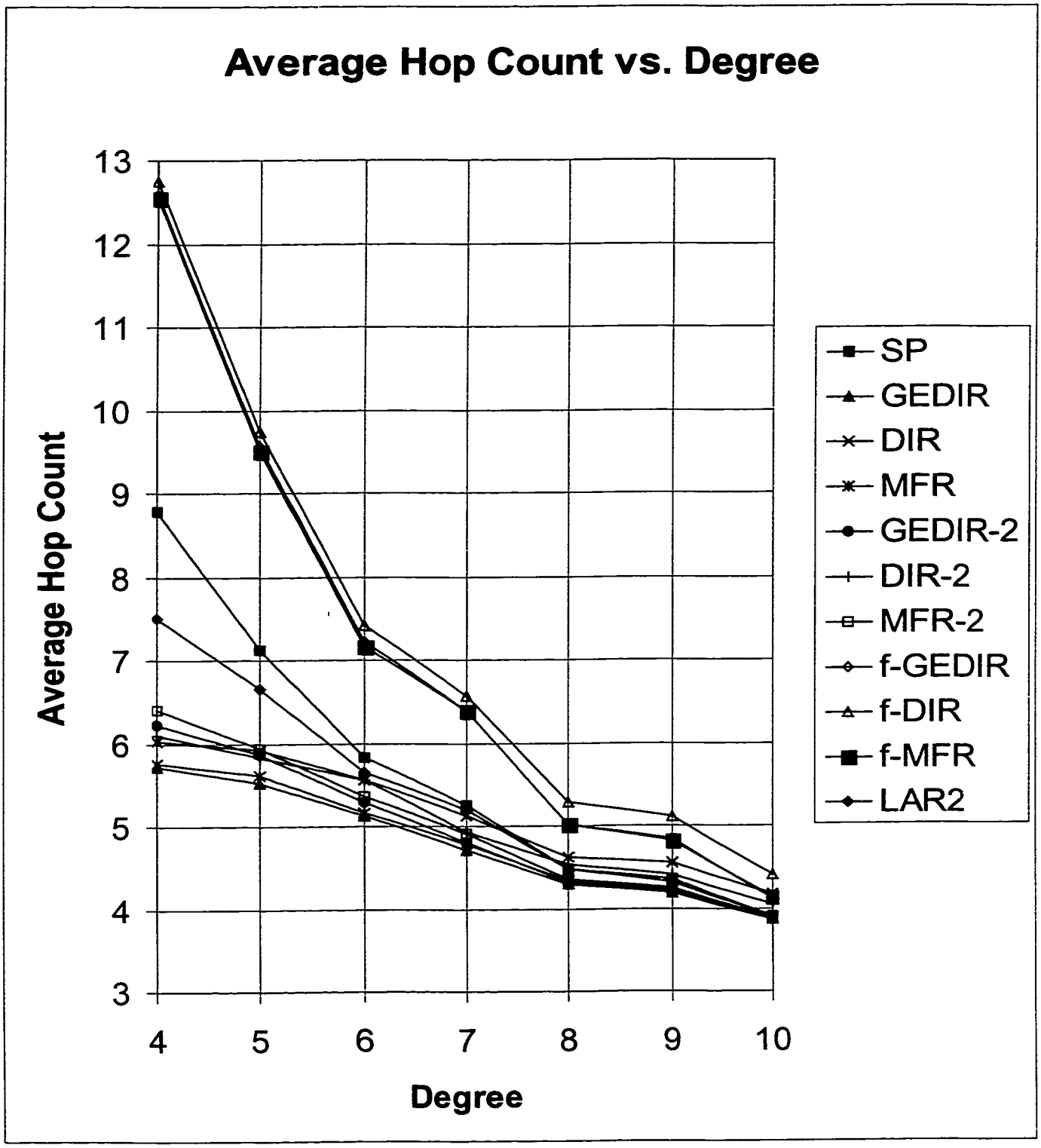
5.3.2 Average hop count for individual methods

Table 5-3 (and Graph 5-2) present average hop counts for methods studied. They are calculated as the sum of hop counts for all the successful transmissions over total number of successful transmissions, for each individual method, thus if a method fails in a transmission, the hop count is given a value of zero. For methods in which a message can be delivered several times (such as flooding variants, multiple-path variants, etc.), the path with shortest hop count is considered. Hence, to determine the hop count for a flooding variant of a basic method, the shortest successful routing path that was discovered by the flooding method is considered. In case of failure of

delivery, the hop count is again given a value of zero. SP method does not give smallest numbers in the table because it provides longer paths in cases where other methods fail. The hop counts for *DIR* based methods are consistently (but not significantly) higher than those of *GEDIR* and *MFR* methods are. Similar results were obtained for other cases. *GEDIR* and *MFR* methods have shown consistently close success rates and hop counts in all cases. The differences in both the success rates and hop counts were less than 1% on the average, with no difference for many of graphs considered. When there was a difference, it appears that one of them was a 'winner' by a random choice, with slight overall advantage in favor of *GEDIR* method. A closer analysis reveals the reason why the path selected by *GEDIR* and *MFR* methods were identical in more than 99% of cases. Consider Figure 3-3. Let A and B be two different nodes selected by the *GEDIR* and *MFR* methods, respectively, when packet is to be forwarded from node S. Suppose that they are located on the same side of SD. $|AD|<|BD|$, since *GEDIR* selects A. Node B cannot be selected within triangle SAA' where A' is the projection of node A on direction SD, since B has more progress than A. However, the angle SAB is then obtuse, and $|SB|>|SA|$. Since A and B are likely to be close to each other, the remaining path may coincide, or at least the chances for delivery are similar. However, when A and B are on the opposite sides of SD, then a difference in success or hop count is more likely bigger.

Degree	4	5	6	7	8	9	10
SP	8.78	7.12	5.82	5.25	4.48	4.35	3.90
GEDIR	5.72	5.53	5.13	4.72	4.29	4.19	3.87
DIR	6.03	5.92	5.55	5.13	4.63	4.55	4.17
MFR	5.75	5.61	5.16	4.78	4.33	4.23	3.89
GEDIR-2	6.23	5.86	5.29	4.81	4.31	4.20	3.87
DIR-2	6.10	5.82	5.55	4.92	4.54	4.41	4.05
MFR-2	6.40	5.93	5.36	4.91	4.36	4.24	3.88
f-GEDIR	12.59	9.55	7.22	6.39	5.01	4.83	4.12
f-DIR	12.75	9.74	7.42	6.57	5.28	5.11	4.41
f-MFR	12.55	9.50	7.17	6.38	5.01	4.82	4.14
LAR2	7.51	6.65	5.65	5.20	4.47	4.31	3.90

Table 5-3. Hop counts for n=100 nodes



Graph 5-2. Hop counts for n=100 nodes

Another interesting fact is that the average hop counts for flooding method, take *GEDIR* as example, are much higher than their basic counterparts. The success rate of basic *GEDIR*, at d=4,

is 49.7% while the *f-GEDIR* is always 100%. Since for each trial case, same pairs of source-destination nodes are run under different protocols, thus 49.7% of the time the *f-GEDIR* method did find concave nodes, thus no flooding required. So in the 100% - 49.7% = 50.3% of the time the average hop count for the *f-GEDIR* is (12.59 – 5.72 x 49.7%) / 50.3% = 19.38. Since the statistics show that number of hops required for a successful basic *GEDIR* algorithm is very similar to that of *Shortest-Path*, thus we can assume that *GEDIR* uses same number of hops as *Shortest-Path*. Based on this assumption, then for the other 50.3% of the time under basic *GEDIR* would had used (8.78 – 5.72 x 49.7%) / 50.3% = 11.8 hops in average, if it had successfully delivered the message. Comparing to 19.38 average hops in *f-GEDIR*, the difference is 7.58 extra hops. If we compare the hop count between *f-GEDIR* and *Shortest-Path*, then 50.3% of time *f-GEDIR* would had used (12.59 – 8.78) / 50.3% = 7.57, almost the same as the extra steps needed for *f-GEDIR* when basic *GEDIR* has failed, considering the number rounding in the calculation. Since the *f-GEDIR* only floods at concave nodes, so when a concave node is reached, the basic *GEDIR* stops right there while the flooding *GEDIR* continued by flooding ALL the neighbors of the concave node. Thus, two transmissions were wasted when the message was sent back to the original neighbor of the concave node, plus another hop to send the message to the next host where basic *GEDIR* can possibly route the message to its destination. Therefore in total 2 extra hops were used to reroute the message back to a possible successful path.

Another case where the *f-GEDIR* would have more hops is when the routing got of a wrong start. Consider the figure in the following page:

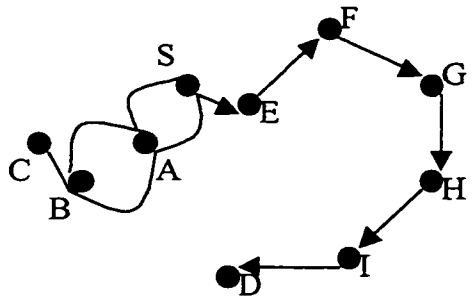


Figure 5-1. A route under *f-GEDIR* from *S* to *D*

At first, the message was transmitted to node A because its geographic distance is closest to node D among all neighbors of S. Then the next hop is node B. At this point since node B is concave node, thus it floods all its neighbor (node A and node C here), so the message was sent back to node A. Node C also receives the transmission. At this stage, another possible path was possible from node C, and node B had become inactive, thus the best neighbor of node A now is node S, since node S is the only reachable neighbor of node A right now. And again at node S it wanted to send the message to node A again, but since the message was transmitted from the same node, thus S becomes a concave node now and floods node A and E and node S becomes inactive. At this point all the neighbors of node A (node B and S) had become inactive, thus transmission at node A ends there. However, the message at node E continues its route to node F, then node G, H, I and eventually reaches its destination: node D. For this scenario, the extra hops required to reroute the message back to the right path is 4 steps: S->A->B->A->S. The message at node C has no possible receiving node since the only neighbor of node C is node B that had become inactive already. In some other cases, it took the algorithm more than just 2 retransmissions before it discovered that the message was sent on the wrong path. Therefore, more extra hops by the flooding from concave nodes were needed to finally redirect the message to the right path. As a side effect, this increases the flood rate in the system.

Compared to flooding variances, the *LAR2* uses much less hops because in *LAR2* there is no 'back-fire', which is caused by bad choice of neighbor resulting concave node's neighbors being flooded to 'bring back' the message to the correct route. Again, as the degree of connectivity increases, the difference in hop counts between *LAR2* and other flooding variances decreases. This is caused by the increased success rate of basic methods, which in turn translates into less 'flooding' in flood variances, thus all methods use more or less similar amount of retransmission (of message).

Counting the hop counts in flooding methods (and multiple path methods) could be little bit tricky, since the destination node could have received more than one copy of message, depending on the flooding nature of the routing protocol. Thus, the hop count for those methods is measured as the minimum number of transmission required for any copy of the message to reach the destination node from the source host. This is the same as finding the path that has least hop counts among all the successful routes that had been used to deliver the message in the network.

In multiple path methods, the routing start by several different possible paths (depending on the value of c), but all those paths might not necessary all reach their destination. Several scenarios could have caused the message on some path to be rejected, thus terminating that route. When a message had reached a concave node the transmission was dropped because no further retransmission is possible; or when the next neighbor to receive the message had already transmitted the same message on another path, thus refuse to send the same message again... This is the same for flooding methods, since each host in the network has a list of message *id*s that they had already transmitted to avoid further flooding the system.

5.3.3 Success rate and average hop count for grouped methods

Degree	4	5	6	7	8	9	10	11	12	13	14
SP	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
1-Hop	55.20%	64.20%	78.80%	87.63%	90.00%	91.90%	97.20%	97.40%	99.40%	99.60%	99.50%
2-Hop	55.60%	64.60%	78.60%	81.25%	82.70%	87.50%	92.20%	89.60%	93.00%	95.60%	93.50%
flooding	84.20%	90.40%	96.80%	99.38%	99.70%	99.60%	99.90%	99.80%	100.00%	100.00%	100.00%

Table 5-4. Grouped success rate for n=100 nodes

Degree	4	5	6	7	8	9	10	11	12	13	14
SP	8.783	7.116	5.816	5.254	4.479	4.3465	3.9	3.681	3.377	3.264	3.0215
1_GEDIR	5.5335	5.3855	5.046	4.66	4.2405	4.1745	3.8575	3.709	3.3855	3.257	3.0355
1_DIR	5.7055	5.6665	5.378	4.9975	4.5485	4.501	4.15	4.047	3.716	3.5765	3.33
1_MFR	5.543	5.4065	5.06	4.682	4.271	4.1935	3.879	3.732	3.41	3.272	3.057
2_GEDIR	5.507	5.2915	4.9625	4.4435	4.112	4.007	3.737	3.526	3.2745	3.161	2.944
2_DIR	5.8585	5.712	5.4235	4.861	4.489	4.384	4.0445	3.8555	3.586	3.474	3.1855
2_MFR	5.5135	5.2995	4.9775	4.475	4.136	4.0235	3.752	3.541	3.288	3.169	2.9575
f_GEDIR	9.4685	8.2395	6.6635	6.1615	4.911	4.7295	4.0885	3.775	3.4685	3.343	3.0595
f_DIR	9.714	8.4775	6.9305	6.358	5.187	5.0065	4.3805	4.107	3.789	3.6505	3.354
f_MFR	9.474	8.221	6.625	6.1545	4.9125	4.719	4.1095	3.797	3.4975	3.3465	3.077
LAR2	7.496	6.649	5.636	5.198	4.4655	4.312	3.897	3.683	3.378	3.2615	3.022

Table 5-5. Grouped average hop counts for n=100 nodes

The success rate for 1-hop methods (in Table 5-4) when all three basic *GEDIR*, *DIR* and *MFR* had successfully delivered the message is slightly better than the group success rate for 2-hop variance. Even though *2-hop GEDIR* and *2-hop MFR* both have improved their basic methods, the lower success rate of *2-hop DIR* which is lower than the basic *DIR* method has caused the lower grouped success rate.

Numbers in Table 5-5 show the average hop counts when ALL methods within each group (basic group, 2-hop group and flooding group) succeeded in delivering the message. The average hop counts of *GEDIR* and *MFR* methods are all very close in both their basic form, 2-hop and

flooding variance while *DIR* usually requires more hops in all the groups. A more careful inspection shows that 17%~34% of time basic *DIR* uses one more hop than basic *GEDIR* method when messages arrive their destinations under both algorithms. The value is obtained as follows:

In the simulation, 20 graphs are generated for each d, and for each graph 100 testing cases are run, thus in total 20 x 100 = 2,000 pairs of source-destination nodes. Since the group success rate for basic methods for d = 4 is 55.2%, thus 2,000 x 55,2% = 1104 successful cases. Since the average hop count is calculated as the total accumulated hop counts (when successful) over total number of successful cases, so the total accumulated hop count for *DIR* is 1104 x 5.7055 = 6299 and for *GEDIR* is 1104 x 5.5335 = 6109. So in total *DIR* had used 6299 – 6109 = 191 more retransmissions than *GEDIR*. Assuming that *DIR* uses at most one more hop than *GEDIR* in all the cases, then we can conclude that 191 / 1104 = 17% of the time *DIR* has longer delay than *GEDIR*.

The difference seems to had jumped from 17% to 28% at from d=4 to 5 and then become more consistent (around 30%) afterward.

When compared to the shortest path algorithm, (1-hop) *GEDIR/MFR* methods have shown encouraging results (taking into account that they are just basic methods that involve no flooding effect). Their success rate for n=*20* nodes was about 67% for d=2, 81% for d=*3, 89%* for d=*4, 94%* for d=*5*. For n=*50* the success rate was about 69% for d=4, *80%* for d=*5, 87%* for d=*6, 91%* for d=*7* and *94%* for d=*8*. The hop counts for *GEDIR/MFR* are comparable to hop counts in *SP*.

5.3.4 Flooding ratio for individual methods

Degree	4	5	6	7	8	9	10
SP	1	1	1	1	1	1	1
GEDIR	0.56	0.70	0.83	0.87	0.93	0.96	0.99
DIR	0.59	0.76	0.91	0.95	1.01	1.04	1.07
MFR	0.57	0.72	0.84	0.89	0.95	0.97	0.99
GEDIR-2	0.62	0.77	0.88	0.90	0.95	0.96	0.99
DIR-2	0.58	0.72	0.87	0.88	0.96	0.97	1.01
MFR-2	0.65	0.78	0.90	0.92	0.96	0.97	1.00
f-GEDIR	4.87	4.46	3.11	2.95	1.96	1.69	1.32
f-DIR	4.72	4.12	3.00	2.62	1.91	1.73	1.39
f-MFR	4.79	4.52	3.03	2.88	1.94	1.66	1.42
LAR2	1.75	2.80	4.34	5.34	6.81	7.96	9.46

Table 5-6. Flooding rates for n=100 nodes

Table 5-6 shows flooding rates for each method for n=100 nodes. Both successful and unsuccessful deliveries are considered. Numbers less than 1 in many cases are obtained because concave nodes are detected before the messages reached their destination, so hop count is smaller compared to *SP* method for the same routing tasks. In order to provide fair comparison with *LAR2* method, all nodes in flooding methods were assumed to memorize past traffic and do not forward the same message twice. This modification had no impact on success rates and hop counts. Flooding based methods, which guarantee 100% delivery (*f-GEDIR* and *f-MFR*), did not significantly flood the network with higher degrees (<2 for d=8, 9, 10; between 5 and 10% of nodes are flooded). However, the effect was notable for lower degrees (>4 for d=4 and 5; up to 40% of nodes were flooded). *LAR2* method had the reverse effect. The flooding rate increased significantly with the degree (from about twice of *SP* at d=4 or 15% of nodes to >9 at d=10 and about 14 at d=14 or over 40% of nodes). Without memorizing messages, the flooding rates of *LAR2* would be much higher (they would increase $O(d^2)$ times). Let us compare *LAR2* methods with flooding based ones. *f-GEDIR* and *f-MFR* methods guarantee delivery, require less memory (only concave nodes need to memorize messages), and have significantly lower flooding rates at moderate and high degrees (from d=6 for n=100). *LAR2* has lower hop counts, but the difference is

significant only for small degree networks. Thus our flooding based methods are superior to *LAR2* for higher degree networks, while guaranteed delivery offers satisfactory compensation for higher flooding rate for lower degree networks. We have also measured how many neighbors of destination would deliver message to it, and established that the number is 1 or very close to 1 for all methods except for *LAR2*, for which that number is > *d/2*.

5.3.5 Analysis on multiple-path methods

Table 5-7 presents experimental results on delivery rates of multiple path methods for n=100 and d=6. The improvements obtained by adding multiple paths are notable, but less than anticipated. The success rate increases by about 3-5% from c=1 to c=2, by additional 2% from c=2 to c=3, and by 1% from c=3 to c=4. Alternative methods have about 5% higher success rates than original ones for all c values. Disjoint methods have about 15~17% better success rate than the corresponding original ones, for all values of c. Similar results were obtained for n=100 and d=4, 5, and 7. It is worth to note that disjoint methods achieve almost same success rate as *LAR2* even at c=1, and involve almost no unnecessary flooding.

Table 5-8 presents hop counts for multiple path methods. Alternate methods have slight hop count increase while disjoint methods used, in average, about one extra hop, compared to original methods. Table 5-9 gives the corresponding flooding rates, with numbers around c, which is expected.

C Value	1	2	3	4
SP	100.00%	100.00%	100.00%	100.00%
orig. GEDIR	77.30%	80.70%	81.95%	82.70%
orig. DIR	79.10%	81.60%	83.00%	83.90%
orig. MFR	78.40%	81.70%	83.00%	83.70%
alt. GEDIR	80.70%	86.05%	87.65%	88.10%
alt. DIR	82.85%	86.95%	88.65%	89.10%
alt. MFR	81.70%	86.55%	87.85%	88.35%
disj. GEDIR	92.10%	96.20%	97.55%	97.80%
disj. DIR	90.90%	95.10%	96.90%	97.30%
disj. MFR	92.25%	96.10%	97.75%	98.00%

Table 5-7. Delivery rates for multiple path methods for n=100 and d=6

C Value	1	2	3	4
SP	5.816	5.816	5.816	5.816
orig. GEDIR	5.1285	5.173	5.1985	5.2105
orig. DIR	5.5515	5.454	5.4735	5.4885
orig. MFR	5.161	5.1995	5.227	5.2465
alt. GEDIR	5.411	5.5535	5.6035	5.596
alt. DIR	5.947	5.9295	5.932	5.8995
alt. MFR	5.444	5.5665	5.573	5.576
disj. GEDIR	6.447	6.2055	6.087	6.057
disj. DIR	6.628	6.303	6.232	6.1925
disj. MFR	6.348	6.134	6.093	6.0525

Table 5-8. Hop counts for multiple path methods for n=100 and d=6

C Value	1	2	3	4
SP	1	1	1	1
orig.c_GEDIR	0.833	1.1905	1.4345	1.6225
orig.c_DIR	0.909	1.2035	1.459	1.645
orig.c_MFR	0.8445	1.214	1.4635	1.6545
alt.c_GEDIR	1.0425	2.171	3.2345	4.0165
alt.c_DIR	1.1375	2.4175	3.5655	4.3875
alt.c_MFR	1.048	2.1925	3.253	4.0555
disj.c_GEDIR	1.1625	2.481	3.6375	4.522
disj.c_DIR	1.194	2.45	3.59	4.381
disj.c_MFR	1.142	2.4555	3.608	4.4935

Table 5-9. Flooding rates for multiple path methods for n=100 and d=6

Chapter 6. Performance evaluation of power efficient algorithms

In this chapter, we would discuss simulation experiments run to compare the performance of *power efficient* algorithms and other constant metric algorithms (including *GEDIR*).

6.1 Experiment parameters for power efficient routing algorithms

In this experiment, the *power efficient, GEDIR, DIR, MFR* algorithms are considered for analysis. Also included are the *2-hop* variants of the constant metric algorithms (*GEDIR, MFR* and *DIR*).

Like previous set of experiment, this one is also carried out using random unit graphs. Each of n nodes is chosen by selecting its x and y coordinates at random in the interval $[0,m)$. In order to control the average node degree k (that is, the average number of neighbors), we sort all $n(n-1)/2$ (potential) edges in the network by their length, in increasing order. The radius R that corresponds to chosen value of k is equal to the length of $nk/2$-th edge in the sorted order. Generated graphs that were disconnected are ignored. We have fixed the number of nodes to $n=100$, and average node degree k to 10. We have selected higher connectivity for our experiments in order to provide for better delivery rates and hop counts, so we can concentrate our study on power conserving effects.

The comparison of *DIR* (compass routing), *MFR* and *GEDIR* methods in [SL1] did not depend on the size m of square containing all the points. However, in case of power consumption, the actual distances greatly impact the behavior of algorithms. More precisely, the path selection (and the energy for routing) in our power saving algorithm depends on the actual size of the square. Although the path selection in *DIR, MFR* and *GEDIR* methods are independent on m, the energy needed for routing differs, and is not simply proportional to m. We compared all methods for squares of sizes m=5, 10, 20, 50, 100, 200 and 500 for the first power model, and m=10, 100, 200,

500, 1000, 2000, 5000 for the second one. The results are averages over 20 graphs with 100 routing pairs in each chosen at random.

6.2 GPS based algorithms considered in the experiment

In our comparisons, the power consumption (cost, power-cost, respectively) in all compared methods was measured by assigning the appropriate weights to each edge. The shortest weighted path algorithm was used as a benchmark. Our comparison for the category of power (only) consumption involved the following GPS based distributed algorithms: *NFP*, random progress, *MFR*, *DIR*, *GEDIR*, *NC*, the proposed distributed power efficient routing algorithm, and the benchmark shortest (weighted) path algorithm (*SP*). The *NFP* is a modified version of *MFP* where instead of choosing the *Most Forward Progressive*, the *Nearest Forward Progressive* neighbor is chosen. In *Random* progress algorithm, equal probability is given to all the forward progressive neighbors when selecting next recipient to forward the message. In case of non-existence of forward progressive neighbor, the backward-progressive node closest to the current node is chosen. A node, currently holding the message, will stop forwarding the message if the best choice, by the corresponding method, is to return the message to the node where the message came from. Such nodes are called concave nodes [SL1], and delivery fails at such nodes.

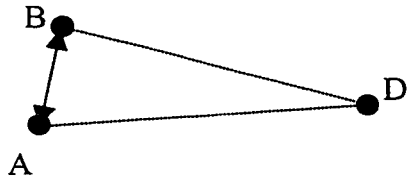


Figure 6-1. *NFP* method fails

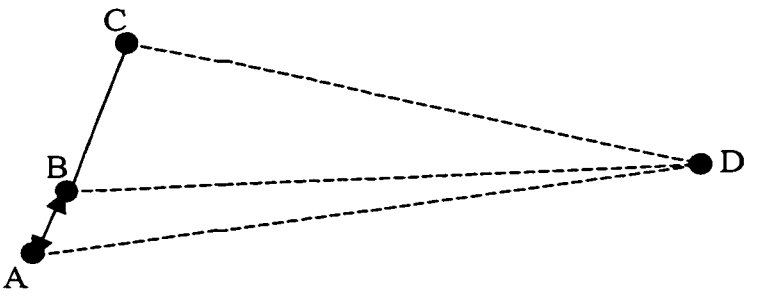


Figure 6-2. *Power1* frequently fails

We have introduced a new routing method, called Nearest Closer (*NC*), in which node *A*, currently holding the message, forwards it to its nearest node among neighboring nodes which are closer to destination than *A*. This method is an alternative to the *NFP* method that was experimentally observed to have very low success rate (fewer than 25% in our case with second power formula). The reason for low success rate seems to be the existence of many acute triangles. Here we have an acute triangle *ABD* (see Figure 6-1) so *A* and *B* are closest to each other, and therefore selected by *NFP* method which then fails at such nodes (*D* is the destination).

The proposed power efficient method, which will be referred as *Power1* method, was also experimentally shown to have very low success rate, especially for large m (under 60% for m>500; under 3% for m>5000). Consider a scenario in which *Power1* fails (see Figure 6-2), where $|AD| < |BD| < |CD|$. Node *A* sends message to closest neighbor *B*. Since *A* is very close to *B* but *C* is not, power formula applied at *B* selects *A* to send message back, and a loop is created. Such case happens more frequently as the square size m increases, due to the power function where the variable r becomes more dominant with larger value. The power efficient algorithm is therefore modified to increase its success rate in the following fashion: only neighbors that are closer to destination than the current node are considered. The modified power efficient algorithm is referred as *PowerM0* in the simulation.

We included 2-hop *GEDIR, DIR, MFR* and *NC* methods in our experiments. 2-hop *NC* method is defined as follows. Each current node *C* finds the neighboring node *A* whose 1-hop nearest (closer to destination *D*) neighbor *B* has the shortest distance (between *A* and *B*). If no such node exist (i.e. none of neighboring nodes of *C* has forward neighbor) then take the neighbor node *E* whose backward nearest neighbor *F* has smallest distance (between *E* and *F*).

6.3 Performance Evaluation

Due to limited space allowed in this section, only statistics accumulated during the simulation run using second power model (E = 500) were discussed. However, since the power formula (and probably associated square size *m*) is the only difference between the simulations, thus the analysis is representative for other power models as well.

6.3.1 Success rate for individual methods

Success/success rate	5	10	20	50	100	200	500
SP-Power	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
SP	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
PowerM0	95.18%	94.81%	95.12%	95.06%	94.71%	94.93%	94.71%
GEDIR	97.37%	97.25%	97.42%	97.33%	97.12%	97.20%	97.26%
DIR	97.89%	97.77%	97.89%	97.72%	97.67%	97.76%	97.81%
MFR	97.71%	97.46%	97.64%	97.51%	97.40%	97.54%	97.60%
NC	94.73%	94.39%	94.37%	94.41%	94.36%	94.23%	94.52%
Random	97.37%	97.43%	97.60%	97.38%	97.38%	97.49%	97.44%
GEDIR2Hop	98.94%	98.78%	98.98%	98.88%	98.80%	98.84%	98.74%
DIR2Hop	91.00%	91.42%	91.05%	91.09%	90.87%	90.92%	90.69%
MFR2Hop	99.40%	99.39%	99.44%	99.39%	99.35%	99.34%	99.30%
NC2Hop	98.36%	98.02%	98.26%	98.25%	98.00%	98.11%	98.13%
NFP	24.04%	23.80%	24.04%	23.84%	23.80%	23.23%	23.42%
Power1	97.36%	97.20%	97.24%	96.96%	96.17%	94.48%	58.62%

Table 6-1. Success rate with Power Formula using E=500.

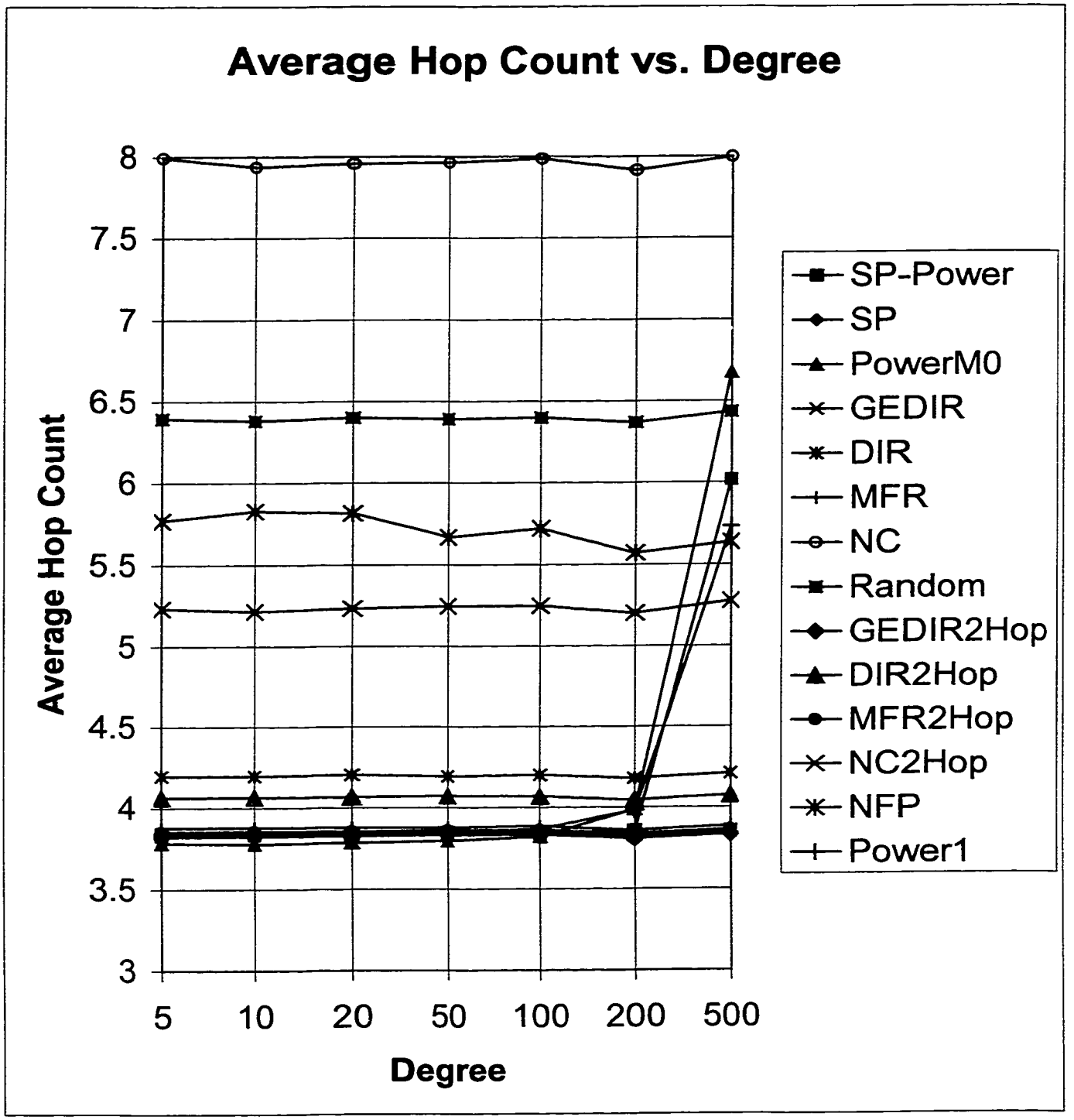
As shown in Table 6-1, the delivery rate for *1-GEDIR, 1-DIR,* and *1-MFR* methods in our experiments were about 97%. *1-NC* had 95% success rate, *2-GEDIR* and *2-MFR* about 99%, *2-DIR* about 91%, *2-NC* and random methods about 98%, and power method 95% success rate. While all other methods choose the same path independently on *m* and power formula applied, power method does not, and almost constant and good delivery rate for it is a very encouraging result. It is noted that *1-NC* has the lowest success rate among all the 1-hop distributed algorithms

(3~4% lower). Also, as already discovered in previous set of experiment, again success rate in *2-DIR* has not improved compared to other 2-hop algorithms.

6.3.2 Average hop counts for individual methods

Method/hops	5	10	20	50	100	200	500
SP-Power	3.8281	3.83225	3.82805	3.83725	3.83815	3.85855	6.017
SP	3.8281	3.83225	3.82805	3.83725	3.83815	3.8154	3.84005
PowerM0	3.7835808	3.77687625	3.78521715	3.7953888	3.81975425	3.98575545	6.6762636
GEDIR	3.8436561	3.8445769	3.8460468	3.8486584	3.8503171	3.82524795	3.85682735
DIR	4.1955368	4.19601735	4.20302795	4.19246455	4.1999846	4.17886275	4.2083791
MFR	3.87601795	3.87879635	3.87676225	3.8779036	3.88272505	3.8576067	3.88456895
NC	7.99271735	7.9369696	7.956985	7.9628875	7.9879195	7.91578675	7.9987016
Random	6.39443615	6.38003775	6.40053665	6.39148435	6.397777	6.3679956	6.4327182
GEDIR2Hop	3.823109	3.8254879	3.8268414	3.8338012	3.83314435	3.8088108	3.83476575
DIR2Hop	4.06271235	4.0653696	4.06526245	4.0716335	4.06708055	4.04371265	4.0747956
MFR2Hop	3.85141555	3.85502125	3.8527864	3.85914475	3.86019015	3.83789205	3.8628713
NC2Hop	5.23192325	5.2144157	5.2328947	5.24587105	5.2467227	5.201169	5.27595215
NFP	5.76783725	5.82799015	5.81333255	5.66825465	5.72039425	5.56986665	5.6366964
Power1	3.84345055	3.84263675	3.8431841	3.8472262	3.8646362	3.98195025	5.7337587

Table 6-2. Hop counts with Power Formula using E=500



Graph 6-1. Hop counts with Power Formula using E=500

The hop counts (in Table 6-2 and Graph 6-1) for no-power based methods are very consistent no matter what the square size m is: 3.9 for *1-GEIDR* and *1-MFR*, 4.2 for *1-DIR*, 7.9 for *1-NC* and

6.4 for *Random*. It is obvious that *1-NC* uses more hop count than all other GPS based methods, which is due to the applied algorithm that chooses closer neighbor than other methods. Hop count for power method increases with larger square size. Clearly, with increased energy consumption per transmission, power method reacted by choosing closer neighbors, resulting in higher hop counts. The turning point after which it becomes obvious that power method is using higher hop count is when the distance-dependant transmission power consumption exceeds reception power at each transmission. For second power formula where E=500, the transmission power is $(E+ d^2)$ and reception power is E. Thus, the ratio is $1+ d^2/E$. The variable term is d^2/E, and substituting E by 500 it becomes $r=d^2/500$. The turning point is where $r>1$, thus $d>22.36$, which is very close to the average edge length (25.29) of the network at square size $m=200$. It is also noted that at this square size the hop count for power method starts to increase.

6.3.3 Average power consumption for individual and grouped methods

The energy consumed in each GPS based method (with routing decisions made independently on power consumption) can be roughly estimated as $h(ad^{\alpha}+bd+c)$, where h is average hop count and d is average edge length in the method. Let us show the superiority of *GEDIR* method over *MFR* method and superiority of compass routing over random progress method.

Let A and B be the nodes selected by the *GEDIR* and *MFR* methods, respectively, when packet is to be forwarded from node S (see Figure 6-3). Suppose that B is different from A (otherwise the energy consumption at that step is the same). Therefore $|AD|<|BD|$. Node B cannot be selected within triangle SAA' where A' is the projection of node A on direction SD, since B has more progress than A (here we assume, for simplicity, that A and B are on the same side of line SD). However, the angle SAB is then obtuse, and $|SB|>|SA|$. From $|SB| > |SA|$ and $|BD| > |AD|$ and

monotonicity of power consumption function, it follows that the packet requires more energy if forwarded to B instead of A. Note that the presented scenario is for an average case. It is possible to find a counterexample showing the opposite (or showing that a path via B exists while there is no path via A) in some very special cases.

Suppose now that A and B are selected neighbors in case of compass and random progress routing algorithms, (we shall use the same Figure 6-3). Since the lengths $|SA|$ or $|SB|$ are not considered when selecting the neighbors, on the average we may assume that $|SA|=|SB|$. However, the direction of A is closer to the direction of destination (that is, the angle ASD is smaller than the angle BAD) and thus A is closer to D than B. Therefore, one can expect more energy saving using compass routing algorithms as compared to random progress one. Again, counterexamples may be easily found for particular cases.

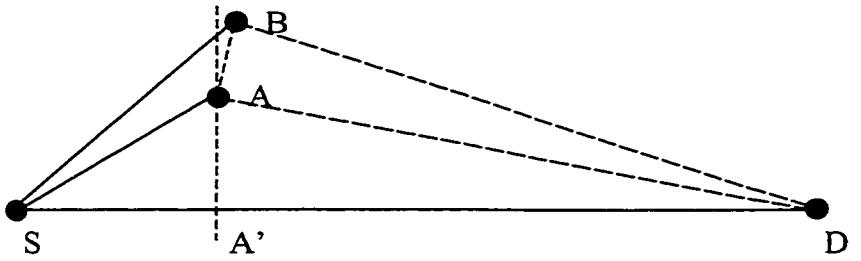


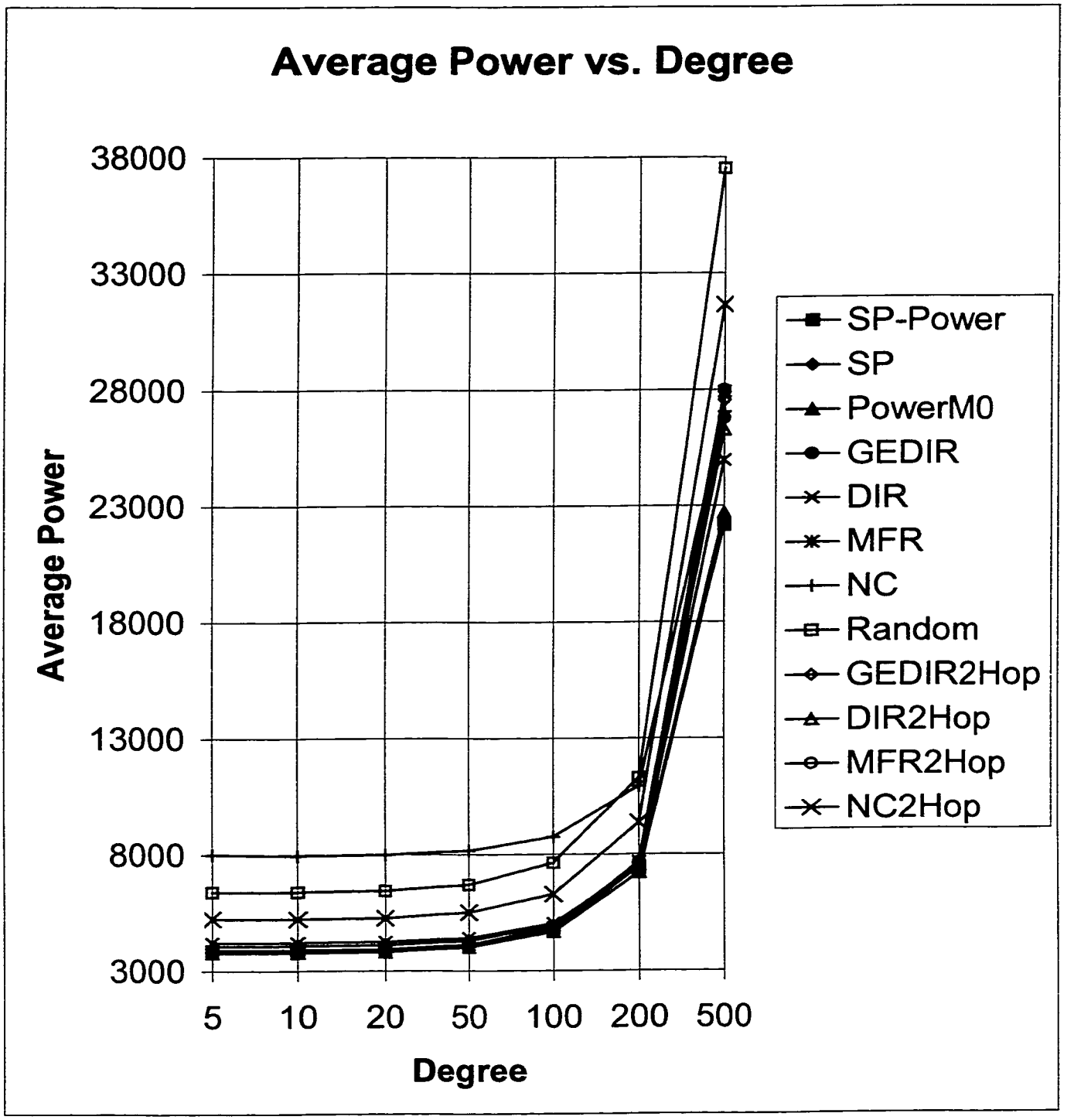
Figure 6-3. GEDIR consumes less power than MFR

Table 6-3 (and Graph 6-2) show the average power assumption (rounded to nearest integers) per routing tasks that were successful by *all* methods, which occurs in about 85% of cases. It is calculated as the ratio of total power consumption (for each method) for these tasks over the total number of such deliveries. The second power formula is used (the results for the first formula are very similar). The experiments were carried for 20 random graphs. As suspected, the power consumption is the smallest with power method. The *1-NC* only saves energy at larger square size. In fact, at smaller square size the *1-NC* is the one that consumes the most power among all

algorithms. In general, at smaller square size m, higher hop count would lead to less power consumption, thus it is not surprising to see that *GEDIR* saves more energy than *MFR* since the former has slightly fewer ho count than latter.

Method/power	5	10	20	50	100	200	500
SP-Power	3830.241161	3840.784067	3862.222438	4051.606878	4693.792477	7209.819593	22167.86446
SP	3830.49558	3841.816422	3866.244808	4077.072153	4795.667874	7629.882932	27897.49309
PowerM0	3785.827786	3785.799967	3820.917629	4016.006185	4680.052086	7218.338819	22706.64549
GEDIR	3845.940742	3853.679853	3882.556177	4077.02948	4762.701941	7458.939214	26786.52266
DIR	4197.603379	4204.25239	4235.990355	4399.275017	5026.813671	7471.713238	24975.52708
MFR	3878.414289	3888.354072	3914.990171	4117.769366	4841.135228	7671.960782	27908.39154
NC	7994.616814	7944.52459	7987.276455	8152.544485	8746.126671	10932.11278	27054.94445
Random	6397.51759	6392.32779	6449.967517	6699.555511	7632.134518	11292.49458	37523.55301
GEDIR2Hop	3825.470201	3834.907212	3864.596398	4070.388447	4777.62936	7565.986899	27557.91804
DIR2Hop	4064.921047	4074.220652	4100.619215	4293.546589	4952.540839	7558.289763	26286.39268
MFR2Hop	3853.817402	3864.607224	3891.181169	4099.89131	4821.442061	7665.966995	27994.59235
NC2Hop	5234.541581	5224.865112	5274.71686	5508.493577	6293.178949	9367.827245	31649.4072

Table 6-3. Power consumption of routing algorithms



Graph 6-2. Power consumption of routing algorithms

The power consumption saving for GEDIR algorithm is better than compass routing method for small values of square size *m*. The reason is that smaller hop count is decisive when no retransmission is desirable. However, for larger *m*, compass routing performs better, since the

greatest advance is not necessarily the best choice, and the closer direction, possibly with smaller advance, is advantageous. The *NC* method is inferior to *GEDIR* of compass routing for smaller values of *m*, because the greatest possible advance is the better choice for neighbor than the nearest node closer to destination. However, for larger values of *m*, *NC* outperforms significantly both, since it simulates retransmissions in the best possible way.

method/power	5	10	20	50	100	200	500
SP-Power	3575.024742	3582.003215	3595.75529	3774.364207	4362.995241	6696.440274	20507.23048
SP	3575.261376	3582.969837	3599.539644	3798.129399	4457.269928	7086.4086	25873.1317
PowerM0	3614.701125	3619.339882	3638.108011	3832.291838	4460.926023	6870.504811	21598.59331
GEDIR	3614.967415	3619.366866	3635.89604	3829.4257	4463.376913	6995.583169	25089.02511
DIR	3929.322196	3931.507327	3953.525886	4115.887342	4686.18299	6968.760821	23297.03922
MFR	3637.682741	3645.387678	3662.139265	3859.796574	4532.883607	7177.251515	26108.05978
NC	7606.977169	7569.159672	7598.837122	7771.294094	8305.833073	10422.43312	25839.37396
Random	5946.620184	5964.766021	5993.282226	6228.509014	7079.364093	10465.06867	34849.44634
GEDIR2Hop	3585.505103	3592.081782	3609.763138	3805.328298	4457.487634	7063.202802	25699.69555
DIR2Hop	3936.896631	3931.763528	3958.399425	4145.004293	4779.734491	7304.429068	25410.87587
MFR2Hop	3598.549685	3606.643769	3623.96737	3822.12097	4482.257916	7131.591155	26014.87484
NC2Hop	4854.337186	4841.688434	4874.31196	5104.494282	5825.806484	8703.336002	29498.95095

Table 6-4. Power Consumption per transmission when all method succeeded

The table above (Table 6-4) further shows that the power consumption per transmission in *power* method was less than that of other localized algorithms, and the difference is very obvious as square size *m* increases. It also proves again that *GEDIR* uses less energy than *MFR*.

As expected, the proposed distributed power efficient routing algorithm outperforms all known GPS based algorithms for all ranges of *m*. For small *m*, it is minor improvement over *GEDIR* or compass routing algorithms. However, for large *m*, the difference becomes very significant, since nearest rather than furthest progress neighbors are preferred. For large *m*, the only competitor is *NC* algorithm. It is also observed that the proposed power efficient algorithm produces paths that

are close to the optimal ones, obtained by *SP-Power* (shortest weighted path) algorithm. More precisely, the overhead (percentage of additional energy per routing task) of power efficient algorithm with respect to SP one is 1.33%, 0.17%, 0.16%, 2.34%, 6.77%, 10.47% and 13.25% for the considered values of m, respectively. Therefore, localized power efficient routing algorithm, when successful, closely matches the performance of non-localized shortest weighted path algorithm.

6.3.4 Path coincidence between selected routing algorithms

An additional fact was also discovered concerning the *GEDIR* and *MFR* algorithms: about 74% of the time both methods used the same routing path. The following table shows the percentage of coincidence between routing path used by different methods.

method/coincidence	5	10	20	50	100	200	500
GEDIR vs. DIR	36.830%	36.565%	36.000%	36.665%	36.405%	36.945%	36.595%
GEDIR vs. MFR	74.120%	73.390%	74.350%	73.610%	73.085%	74.070%	74.205%
GEDIR vs. SP	34.910%	34.890%	34.670%	34.805%	34.615%	34.810%	34.850%
GEDIR vs. Power	94.500%	93.025%	90.085%	81.460%	67.695%	39.230%	7.190%
GEDIR vs. SPPower	39.085%	39.240%	38.905%	38.885%	38.970%	37.630%	9.455%
DIR vs. MFR	30.195%	29.740%	29.200%	29.685%	29.250%	30.200%	29.800%
DIR vs. SP	25.810%	25.270%	25.400%	25.560%	25.670%	25.740%	25.405%
DIR vs. Power	36.495%	36.495%	36.750%	39.850%	44.535%	56.650%	9.155%
DIR vs. SPPower	42.515%	42.345%	41.975%	42.780%	42.800%	43.635%	13.385%
MFR vs. SP	34.485%	34.045%	34.200%	34.170%	34.225%	34.275%	34.570%
MFR vs. Power	71.120%	69.385%	68.260%	61.725%	51.105%	30.930%	6.150%
MFR vs. SPPower	32.910%	32.765%	32.470%	32.525%	32.510%	31.225%	8.055%
SP vs. Power	34.695%	34.645%	34.520%	34.395%	32.665%	26.810%	6.450%
SP vs. SPPower	36.720%	35.895%	36.620%	36.310%	36.415%	34.185%	8.535%
Power vs. SPPower	39.325%	39.720%	40.190%	42.140%	44.955%	51.795%	32.170%

Table 6-5. Path coincidence between different methods.

Chapter 7. Performance evaluation of cost and power-cost efficient routing algorithms

In this chapter, ALL the routing algorithms are considered: constant metric basic and 2-hop (*GEDIR, DIR and MFR)*, *power-efficient*, *cost-efficient* and *power-cost efficient* algorithms. Also added to the experiment environment are some new parameters aiming at creating a more realistic approach to the real-life scenario that involves battery life in the hosts and energy consumption in message reception and transmission.

7.1 Experiment parameters

The performance evaluation in the previous section shows the superiority of power efficient method with nodes that have relatively equal and high battery powers (e.g., all batteries are refreshed). In this section, we shall assume that nodes have different remaining powers. We have also included two more routing methods, cost and power-cost, with their respective shortest weighted path algorithms.

The simulations that evaluate cost and power-cost routing algorithms are designed as follows. Random unit connected graphs are generated as in the previous section. Instead of running 100 routings for each graph, we decided to run iterations (where each iteration is a routing task specified by the random choice of source and destination nodes). Any node currently holding the message will halt the routing task if it is a concave node for that task (method failure), or if the current node does not have enough power to transmit the message to the next node (power failure), or if the recipient node does not have enough power to receive the message (power failure). The iterations are run until the first node in the network 'dies', i.e. does not have enough power to transmit (or receive) the message to the next node (which is decided by the corresponding method).

Each node is initially assigned an energy level at random in the interval [*minpow, maxpow*], where parameters are chosen differently for different values of *m*. After sending a message from node *A* to node *B*, the energy that remained at *A* (and *B*) is reduced by the power needed to transmit (and receive) the message, respectively. The experiment is performed on 20 graphs for each method, for each of two formulas. Note that the suffix of "Methodx" (x range from 0 to 2) represents the different factor *t*: CostMethod0 for $t = f'(A)$; CostMethod1 for $t = f(A)$, etc. To be more specific, below is how the cost and power-cost function are defined for different cost and power-cost efficient routing methods:

Routing Algorithm	Cost/power-cost function used
CostMethod0	$c(A) = f(A) + f'(A)*s/R$
CostMethod1	$c(A) = f(A) + f(A)*s/R$
SPPowerCost1	$pc(A)=f'(S)u(r) + u(r') + f(A)$
PowerCostMethod0	$pc(A)= u(r) f(A) + v(s)f'(A)$
PowerCostMethod1	$pc(A)= f'(S)u(r) + u(r')f(A) + v(s)f'(A)$
PowerCostMethod2	$pc(A)= u(r) f(A) + v(s)f(A)$

Where $f(A) = 1/g(A)$, $g(A)$ denoting the normalized power level of node A; $f'(A) = avg(f(X))$ where X are reachable neighbors of node A; S is the source node; r' is the average length of any edge from S.

7.2 Performance Evaluation

Due to similar reason described in section 6.2, only data collected using power constant E=500 are considered for analysis, but same can be applied to other power constants.

7.2.1 Success rate for individual methods

Because of experience with success rate of power efficient method, we considered two variants of cost and power-cost algorithms. In the modified methods, we restrict the selection of neighbors to the nodes that are closer to destination than current node. The success rates for unrestricted versions (all neighbors considered, methods suffixed with letter "A") were again very low in our experiments (e.g. 40~55% and 28~44% for cost and power-cost methods with different factor *t*, respectively, for *m*=500). So consequently these methods are deemed not viable (the number of

iterations before dying for them is not shown, because high failure rate certainly distorted their interpretation). The success rate for different methods is given in Table 7-1.

method/success	5	10	20	50	100	200	500
SP	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
SP-Power	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
SP-Cost	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
SP-PowerCost	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
SPPowerCost1	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
PowerM0	95.19%	95.07%	95.28%	95.51%	95.54%	95.21%	95.06%
CostM0	92.92%	93.16%	93.00%	93.32%	93.22%	93.18%	92.95%
CostMethod1	92.65%	93.13%	92.95%	93.24%	92.87%	93.00%	92.88%
PowerCost1	91.82%	92.41%	92.50%	93.14%	93.12%	93.45%	93.24%
PowerCostMethod0	91.70%	92.45%	92.57%	93.01%	93.12%	93.43%	93.22%
PowerCostMethod1	91.69%	92.38%	92.52%	93.02%	93.10%	93.40%	93.17%
PowerCostMethod2	92.22%	93.03%	93.04%	93.59%	93.58%	93.54%	93.42%
1-GEDIR	96.80%	96.78%	97.08%	96.94%	97.18%	96.75%	96.85%
1-DIR	97.66%	97.18%	97.76%	97.84%	97.78%	97.54%	97.68%
1-MFR	97.13%	96.90%	97.37%	97.35%	97.46%	97.16%	97.27%
1-NC	94.07%	94.19%	94.70%	94.71%	95.02%	94.21%	94.48%
Random	98.26%	97.81%	98.14%	97.42%	98.13%	97.61%	97.79%
2-GEDIR	98.44%	98.35%	98.61%	98.75%	98.58%	98.33%	98.46%
2-DIR	90.80%	91.37%	90.72%	91.62%	91.69%	91.17%	91.03%
2-MFR	98.80%	98.70%	98.98%	99.07%	98.98%	98.89%	99.04%
2-NC	96.06%	97.45%	97.17%	96.95%	97.44%	96.97%	96.95%

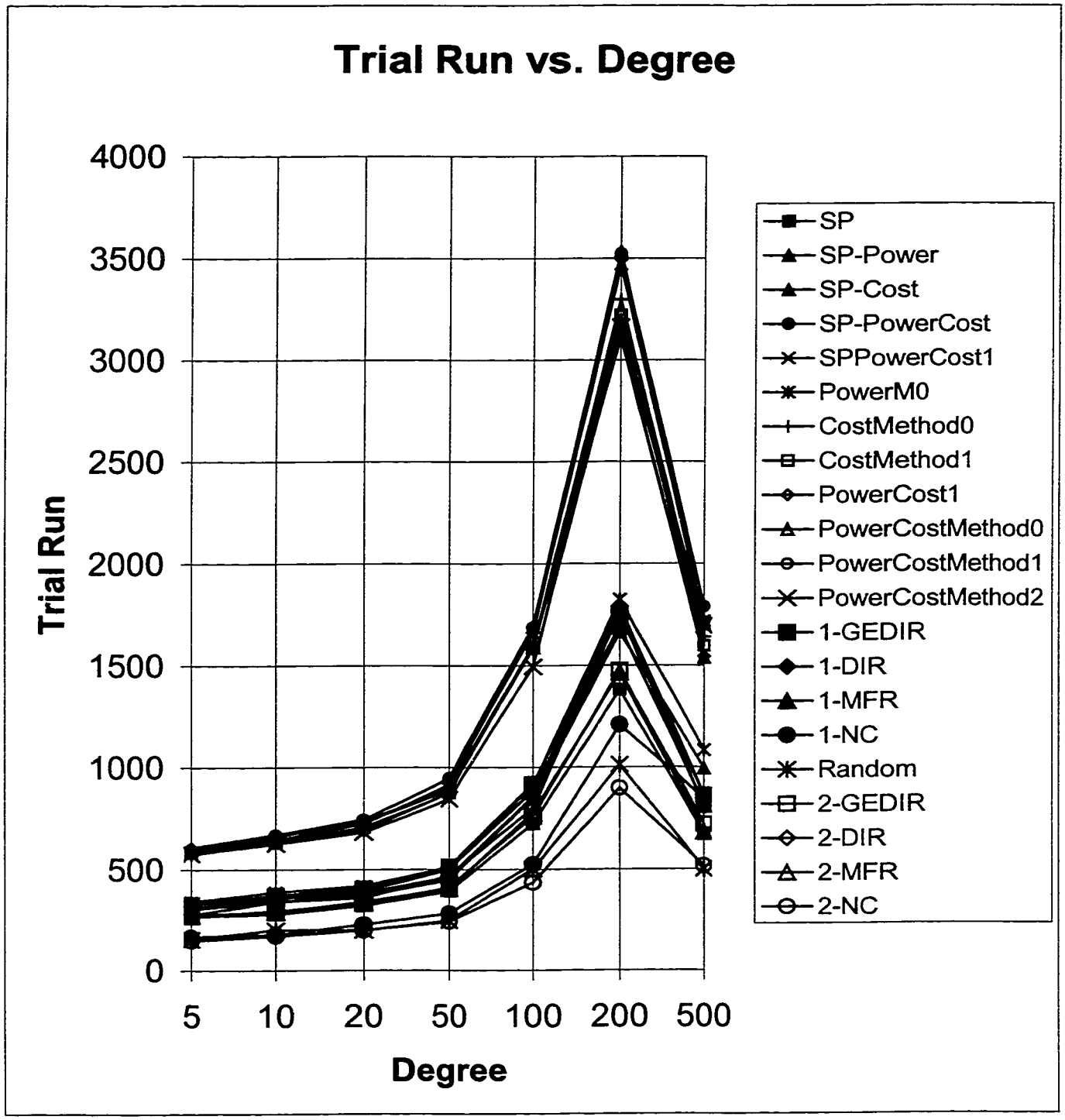
Table 7-1. Success rate for individual methods

We can notice that the GPS based methods with constant metrics usually have a higher success rate (about 2~3%) than all the power aware methods (including power efficient, cost efficient and power-cost efficient methods).

7.2.2 Number of iteration before a node 'dies'

Method/trial count	5	10	20	50	100	200	500
SP	267.7	278.5	320.75	392.7	721.95	1381.2	674
SP-Power	303.95	340.9	371.15	448.1	850.35	1664.3	992.6
SP-Cost	598.1	666.7	742.75	947.6	1688.5	3535	1702.55
SP-PowerCost	598.55	667.75	743.05	947	1685.85	3522.15	1787.35
SPPowerCost1	583.2	640.85	724.1	915.3	1659.4	3468.65	1716.05
PowerM0	339.1	386.75	417.5	511.45	908.8	1824.75	1082.15
CostMethod0	573.15	632.45	692.5	877.8	1584.1	3298.4	1640.6
CostMethod1	586.4	631.4	703.9	874.05	1591	3220.9	1596.65
PowerCost1	605.75	668.5	731.2	890.85	1591.65	3125.95	1542.75
PowerCostMethod0	585.25	659.85	735.1	894.1	1588.25	3115.2	1536.4
PowerCostMethod1	586.25	662.5	729.9	892.25	1592.3	3148.35	1713.3
PowerCostMethod2	577.25	625.75	681.6	844.35	1492.8	3161.95	1694.95
1-GEDIR	327.75	370.05	403.7	512.9	916.65	1753.8	860.45
1-DIR	307.5	357	382.9	462.1	852.75	1788.55	862.6
1-MFR	325.15	369.35	391.25	501.75	882.3	1691.05	814.4
1-NC	166.55	175.25	226.25	281.95	524.1	1208.45	843.8
Random	154	200.1	197	245.5	500.8	1011.75	498.8
2-GEDIR	269.1	293.85	330.15	410.05	758.4	1475.1	720.2
2-DIR	274.1	341.6	360.4	468.4	797.65	1764.2	806.8
2-MFR	274.85	288.9	338	405.3	774.45	1464.6	681.5
2-NC	148.15	170.55	199.1	240.75	434.05	898.15	515.5

Table 7-2. Number of iterations before one of node in each method dies



Graph 7-1. Number of iterations before one of node in each method dies

The number of iterations before a node "dies" for each method is given in Table 7-2 and shown graphically in Graph 7-1. As expected, the cost and power-cost methods usually outlast other GPS

based methods. In fact, the trial runs for cost and power-cost efficient routing methods are almost double (or more) of that of other GPS based methods. Additional observation shows that at square size *m>200*, the *NC* actually extends the network lifetime longer than *GEDIR, MFR* and *DIR*. Similar conclusion can be drawn for power-cost method versus cost method. Correlating to numbers in Table 7-4 that shows the average power consumption per routing, we have also confirmed that after the break-even point, where power consumption for transmission exceeds that of reception, power-cost method outperforms cost method. Same goes for *NC* versus *GEDIR, MFR* and *DIR*.

It should also be noted that sometimes very low success rate contributes likewise to longer trial, because less energy is actually consumed when a routing failed before the destination is reached, compared to a successful delivery. Complete data had shown that the unrestricted versions of routing methods have very high trial count, but also have extremely low success rate. However, since all the methods listed in Table 7-2 have high success rate (over 90%), such factor is negligible in our analysis.

7.2.3 Average hop counts for individual methods

Method/hops	5	10	20	50	100	200	500
SP	3.84637	3.85723	3.874497	3.844546	3.849965	3.867184	3.853569
SP-Power	3.841703	3.863569	3.870272	3.836336	3.850156	3.902505	6.044905
SP-Cost	4.021808	4.025356	4.025524	4.022198	4.02101	4.079662	4.078144
SP-PowerCost	4.022734	4.025017	4.02685	4.024251	4.055835	4.356987	6.252969
SPPowerCost1	3.991482	3.994744	3.999401	3.982571	4.014485	4.092526	4.285754
PowerM0	3.804246	3.829633	3.833429	3.794899	3.846046	4.038902	6.62659
CostM0	4.438631	4.440572	4.427723	4.438342	4.479328	4.554632	4.601365
CostMethod1	4.79848	4.772739	4.756201	4.84179	4.878592	5.02032	5.086877
PowerCost1	5.769305	5.488963	5.141577	4.766416	4.63221	4.835498	6.692544
PowerCostMethod0	6.310357	5.897615	5.428083	4.939612	4.733084	4.990969	6.63813
PowerCostMethod1	6.309151	5.896223	5.420057	4.943082	4.730773	5.030427	7.031812
PowerCostMethod2	5.982685	5.551451	5.143019	4.895716	4.744421	5.006874	5.827857
1-GEDIR	3.850623	3.869549	3.882768	3.830759	3.857324	3.868724	3.875743
1-DIR	4.17385	4.200221	4.233577	4.193122	4.203341	4.225818	4.220682
1-MFR	3.891126	3.899837	3.913325	3.878955	3.895756	3.91389	3.91045
1-NC	8.07005	7.908505	7.946308	7.96049	8.005743	7.974646	8.024264
Random	6.444362	6.461214	6.356028	6.526297	6.49909	6.41358	6.414488
2-GEDIR	3.829179	3.848196	3.854548	3.838494	3.844668	3.856405	3.854449
2-DIR	4.088635	4.079187	4.080202	4.056966	4.094162	4.09038	4.075855
2-MFR	3.860058	3.868562	3.885307	3.846575	3.872109	3.8839	3.882685
2-NC	5.428715	5.231386	5.382343	5.177797	5.31826	5.242866	5.197833

Table 7-3. Average hop counts for individual methods

As explained in section 2.3.2 in previous chapter, the "transmission cost/reluctance" represents the "reluctance" a node to receive and retransmit the message, thus its value does not depend on the distance, and it is proved here in Table 7-3. The average hop count for cost efficient methods is constant no matter what the square size m is, different to higher average hop count for power and power-cost efficient methods with larger square size.

The hop count for all the methods with constant metric is more or less constant, because the same graphs are used during the simulation, only distance between each nodes (and transmission radius) is increased by m. This proves that the performance of algorithms with constant metric is constant as long as the network context does not vary.

7.2.4 Average power consumption per routing task for individual methods

Method/power	5	10	20	50	100	200	500
SP	5771.885	5795.227	5849.341	6000.131	6709.049	9563.507	29146.93
SP-Power	5764.63	5803.746	5838.893	5961.806	6607.527	9159.173	24834.06
SP-Cost	6035.152	6047.82	6077.407	6276.825	7006.265	10038.71	30504.1
SP-PowerCost	6036.541	6047.295	6079.166	6274.555	7011.148	10082.86	27442.24
SPPowerCost1	5989.665	6001.917	6038.161	6212.81	6960.918	9817.047	28059.92
PowerM0	5708.559	5753.234	5785.24	5906.003	6610.586	9249.273	25672.18
CostMethod0	6660.26	6670.148	6678.68	6889.279	7648.324	10560.07	30317.63
CostMethod1	7200.063	7168.544	7171.756	7497.324	8260.579	11315.72	31403.29
PowerCost1	8656.52	8243.445	7750.965	7380.512	7836.801	10626.9	27862.44
PowerCostMethod0	9468.156	8856.607	8181.188	7640.787	7986.947	10836.48	28121.23
PowerCostMethod1	9466.345	8854.521	8169.116	7645.675	7981.341	10858.86	28401.93
PowerCostMethod2	8976.574	8337.118	7752.671	7573.132	8008.416	10967.63	28866.16
1-GEDIR	5778.155	5813.233	5860.025	5966.285	6672.656	9370.093	28108
1-DIR	6262.784	6308.398	6382.91	6490.377	7110.212	9571.907	26565.23
1-MFR	5839.018	5859.12	5907.621	6050.618	6776.848	9629.788	29322.67
1-NC	12106.96	11870.2	11949.43	12127.2	12757.28	14951.93	30786.6
Random	9669.592	9704.025	9582.366	10098.16	10985.56	14493.4	40048.9
2-GEDIR	5746.056	5781.515	5818.757	5987.167	6684.575	9473.785	28797.16
2-DIR	6135.12	6127.428	6154.973	6300.456	7007.036	9607.814	27648.39
2-MFR	5792.413	5812.183	5865.502	6001.47	6741.968	9577.516	29226.01
2-NC	8145.689	7857.234	8114.515	8015.48	9003.151	11904.5	32863.62

Table 7-4. Average Power Consumption per routing

After each method dies (trial run is terminated), the remaining energy level (battery lifetime) of each node in the network was summed up and averaged. It is natural to assume that the method with longer trial run (number of iteration) would have less energy left at the end. However, the reverse is not always true. The numbers displayed in Table 7-5 below shows that before the break-even point, the power-cost efficient methods had higher power consumption than the cost efficient methods. One interesting fact that is worth mentioning is that *NC* always have the lowest average remaining power level than *GEDIR, MFR* and *DIR*, even though its trial run number is much smaller than those methods at smaller square size(see Table 7-2). The explanation can be found in

Table 7-4: at smaller square size, *NC* consumed more energy than other (*GEDIR, MFR* and *DIR*) methods, so even though it has less iteration number in the simulation, but since its power consumption is much higher, thus less remaining power at the end.

7.2.5 Average remaining power level per node for individual methods

Method/power	5	10	20	50	100	200	500
SP	59530.49	68906.07	76307.99	101461.4	201523.6	615355.7	929202.8
SP-Power	57470.91	65258.51	73394.97	98330.47	193793.1	595225.1	880063
SP-Cost	39023.91	44843.72	50067.58	65706.2	132025.3	393380.1	608545.1
SP-PowerCost	38987.15	44782.22	50047.5	65758.23	132106.4	393109.5	636635.1
SPPowerCost1	40184.5	46702.42	51467.22	68290.01	134783.9	407621	645470
PowerM0	56006.69	63249.79	71399.38	95409.62	190872.3	582252.1	853683.3
CostMethod0	37355.94	43364.55	49427.12	65374.22	130268.3	405122.8	638712
CostMethod1	33366.97	40290.25	45117.69	60323.01	120442.3	389285.7	634072.9
PowerCost1	23634.79	30755.24	39203.75	60161.48	126841.2	421127.2	704454.9
PowerCostMethod0	21088.76	27802.28	35935.75	57746.04	124870.2	416217	702262.3
PowerCostMethod1	21023.5	27543.97	36420.2	57842.86	124606	411851.4	648948.6
PowerCostMethod2	24630.86	34104.13	43278.41	62452.59	132850.7	408502	645975.8
1-GEDIR	56140.26	63673.41	71504.89	94557.41	189148.8	584655.3	886650.8
1-DIR	55779.81	62596.74	70732.88	95145.85	189554.2	577196.2	898712.3
1-MFR	56076.18	63487.88	71975.82	94789.95	190375.1	585417.2	888405.2
1-NC	54126.07	63387.06	66949.77	89954.64	180504.6	558703.5	860132
Random	58755.46	64443.34	74301.67	98617.83	191340.4	585763.3	906818.9
2-GEDIR	59544.1	68134.21	75864.89	100523.3	199350.1	608227	919169.8
2-DIR	58478.04	64452.53	73269.78	96146.69	195205.5	582024.4	911184.7
2-MFR	58918.3	68128.03	75051.96	100442.9	197487.1	606333.9	925400.7
2-NC	61490.08	69828.49	77457.48	102854.3	205342	623098.9	936567.5

Table 7-5. Average remaining power per node

Our experiments confirmed the expectations on producing power savings in the network and/or extending nodes lifetime by applying our algorithms on randomly generated unit graphs. The largest number of iterations was recorded for the cost efficient routing method. However, differences are not significant enough to make a conclusion on the choice of best power saving method. The success rates of power, cost, and power-cost methods were in the range of 92-96%.

Conclusion and future work

This thesis described several GPS based localized routing algorithms. Some of them aims at achieving minimal hop count and maximal success rate, while others are intended to reduce power consumption and/or extending lifetime of hosts in the network.

The proposed demand-based distributed algorithms operate in the same manner if some nodes are in the 'sleep' mode. The only modification required is to include a condition at each node to ignore its neighbors that are temporarily not receiving messages. If nodes that are in the 'sleep' mode are actual destinations, the messages for them should be stored until they are ready to receive them.

The obtained experimental results (in chapter 5) show that *DIR* method does perform well in practice, as claimed in [BCSW, KV], and its superiority to non-GPS based methods is therefore not surprising. However, we showed that it can be further improved in various ways. For instance, *DIR* method is not loop-free while *GEDIR* and *MFR* are loop-free. Hop counts for later two methods were slightly better for all graphs, while success rates were comparable. The *GEDIR* and *MFR* algorithms, on the other hand, differed by less than 1% on each metric. If one of them is to be selected, *GEDIR* has a slight advantage in its conceptual simplicity and in using shorter edges on average, which may provide some power savings [SL2] and somewhat fewer transmission conflicts.

Similarly, we have shown overall superiority of flooding based methods (*f-GEDIR* and *f-MFR*) over *LAR2*. They guarantee delivery and require less memorization. Their flooding rates are superior for moderate and higher degree graphs. While flooding rate of *LAR2* is lower for lower degree graphs, its failure rate is also significant for these graphs. Thus, the choice of flooding

based methods even for lower degree graphs is justified by the guaranteed delivery. Full flooding at concave nodes may be replaced by a kind of controlled one, but we were unable to find a way that still guarantees delivery. The advantage of *LAR2* over multiple-path variants might only be in success rate. However, our experiments show that *c-GEDIR* and *c-MFR* may provide multiple paths with comparable success rates and a much smaller flooding rates. Moreover, our experiments clearly show that multiple paths do not add much to success rates. Second path improves success by only about 5%, while each additional path increase their chance of successful delivery by only 1% each. Adding memory seems to have more impact, especially for disjoint methods that achieve similar success rates as *LAR2* even for one-path case ($c = 1$).

Comparing to those GPS based algorithms with constant metric, the power-aware algorithms clearly have the upper hand when comparing the power consumption and/or node lifetime extension. It is also clear that the combined power and cost efficient method: power-cost efficient routing algorithm is the indisputable winner. However, because of the nature of power efficient methods (higher hop count with larger square size), the hop count of power-cost efficient methods is not constant. Therefore, if the design requirement is to achieve the minimum delay, the other constant metric algorithms such as *GEDIR* or *DIR* would be preferred. Moreover, the internal power consumption that is used for applying the algorithm was not considered during the simulation. Since the power consumption is directly proportional to the complexity of the algorithm, thus in some scenario, the constant metric algorithms such as *GEDIR* that requires less computational power might be better choice. The proposed power-aware routing algorithms are all demand-based and can be augmented with some of the proactive or reactive methods reported in literature, to produce the actual protocol. These methods use control messages to update positions of all nodes to maintain efficiency of routing algorithms. However, these control messages also

consume power, and the best trade-off for moving nodes is to be established. Therefore, further research is needed to select the best protocols. One of our primary interests in this thesis was to examine power consumption in case of static networks and provide basis for further study. Our method was tested only on networks with high connectivity, and their performance on lower degree networks remains to be investigated. Based on experience with basic methods like GEDIR [SL1], improvements in the power routing scheme to increase delivery rates, or even to guarantee delivery [BMSU,MS] are necessary before experiments with moving nodes are justified.

The search for distributed routing methods that have excellent delivery rates, short hop counts, small flooding ratios and power efficiency is far from over even for the case of static nodes. 2-hop variants of flooding or multiple path methods may be further studied. Since the battery power is not expected to increase significantly in the future [SWR] and the ad hoc networks, on the other hand, are booming, power aware routing schemes need further investigation. Next, [BMSU] designed a routing algorithm that guarantees the message delivery in unit graphs without the use of any flooding based approach or any memorization technique at the nodes (the same assumptions as in *GEDIR* algorithm). The delivery rate of several routing algorithms is improved in [LS] by ignoring non-intermediate nodes in routing decisions. Finally, [LS] presented routing algorithms that are also suitable for geocasting. Further research is then needed to identify the best GPS based routing protocols for various network contexts. These contexts include nodes positioned in three-dimensional space, obstacles between nodes, nodes with unequal transmission powers, or networks with unidirectional links. Simulations with moving nodes are, of course, the ultimate goal. Experiments with static networks will provide best candidates for the design of routing protocols in mobile networks. The candidate methods that perform best for static nodes shall be combined with

known and novel control messages schemes for location updates to obtain improved GPS based routing protocols.

The formulas for power, cost, and power-cost methods may also need some improvements, and our experiments do not give an ultimate answer on even the selection of approach that would give the most prolonged life to each node in the network. The presented distributed routing algorithms may be improved by multiplying the optimal power (or power-cost) for the remaining transmissions by a factor that will depend on the network conditions. Our energy consumption formulas are easily adjustable to different laws for received power in terms of transmitted power and distance. Also, the corresponding distributed algorithms may be improved by studying different ways of selecting neighbors (that is particularly the case in the power-cost routing algorithm for improving the judgement on whether to send packet directly to destination from the node currently holding it).

It is also an interesting problem to consider power efficient broadcasting, in which a node wants to send a message to all other nodes in MANET. Previous studies concentrated on the time needed for broadcasting or number of retransmissions.

Further study on delivery efficiency can be given to power and power-cost efficient routing methods, since the simulation experiments are done at a static random unit graph network with a high degree of connectivity (and thus high success rate). 2-hop improvement can enhance the delivery rate, and even flooding or multiple-path variants might actually increase or even guarantee the delivery without consuming too much energy.

Finally, the reduction of CPU power consumption, in addition to optimizing transmission power, deserves further investigation. For instance, the Dijkstra's shortest weighted path algorithm runs in $O(n^2)$ time (where n is the number of nodes in MANET), which can be improved to $O(n \log$

n) time using complicated data structures (therefore having high constants and potentially higher time complexity for smaller networks). Is linear time algorithm for computing optimal transmission power routing path possible? Galtier [G] argues that the CPU energy consumption is proportional to the square of the execution time, and therefore significant saving can be achieved by using parallel processing. The conversion of sequential to parallel distributed routing algorithms is straightforward, although addition and deletion of neighbors requires careful design. The design of a parallel shortest weighted path routing algorithm is also justified.

References

[BBCDFPMM] P. Bose, A. Brodnik, S. Carlsson, E. Demaine, R. Fleischer, A. Lopez, P. Morin and J. Munro, Online routing in convex subdivisions, Technical Report, School of Computer Science, Carleton University, 2000 (submitted to ISAAC 2000)

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

[BCS] S. Basagni, I. Chlamtac, V.R. Syrotiuk, Dynamic source routing for ad hoc networks using the global positioning system, Proc. IEEE Wireless Communications and Networking Conference, New Orleans, September 1999.

[BM] P. Bose and P. Morin, Online routing in triangulations, in Proceeding of the Tenth Annual International Symposium on Algorithms and Computation (ISAAC 1999), pp. 113-122, 1999

[BMJHJ] J. Broch, D.A. Maltz, D.B. Johnson, Y.C. Hu, J. Jetcheva, A performance comparison of multi-hop wireless ad hoc network routing protocols, Proc. MOBICOM, 1998, pp. 85-97.

[BMSU] P. Bose, P. Morin, I. Stojmenovic and J. Urrutia, Routing with guaranteed delivery in ad hoc wireless networks, 3rd int. Workshop on Discrete Algorithms and methods for mobile computing and communications, Seattle, August 20, 1999, pp. 48-55.

[CC] C.J. Chang and J.F. Chang, Optimal design parameters in a multihop packet radio networks using random access techniques, in Proc. IEEE GLOBECOM, 1984, pp. 493-497.

[CL] D. Camara and A.F. Loureiro, A novel routing algorithm for ad hoc networks, Proc. HICSS, Hawaii, January 2000, to appear.

[CN] S. Chen and K. Nahrstedt, Distributed quality-of-service routing in ad hoc networks, IEEE J. Selected Areas in Communications, 17, 8, August 1999, pp. 1488-1505.

[CZ] A. Chockalingam and M. Zorzi, Energy consumption performance of a class of access protocols for mobile data networks, Proc. Vehicular Technology Conference, Ottawa, Ontario, Canada, 1998.

[EGHK] D. Estrin, R. Govindan, J. Heidemann, S. Kumar, Next century challenges: Scalable coordination in sensor networks, Proc. MOBICOM, 1999, Seattle, pp. 263-270.

[EMMB] R. Estrada, D. Munoz-Rodriquez, C. Molina, K. Basu, Cellular position location techniques, a parameter detection approach, Proc. 49th IEEE Int. Vehicular technology Conference VTC'99, Houston, May 1999, pp. 1166-1171.

[FFP] K. Fath, P. Flocchini, S. Pierre, A compact routing technique for communication networks, Proc. 1999 IEEE Canadian Conference of Electrical Engineering and Computer Science, 1999, pp. 191-196.

[G] J. Galtier, Using parallel computing to reduce CPU power, Workshop on optimization methods for wireless computing, pp. 32-34, Montreal, Canada, December 1998.

[GFM] J.J. Garcia-Luna-Aceves, C.L. Fullmer and E. Madruga, Wireless mobile internetworking, manuscript.

[GK] P. Gupta and P.R. Kumar, Critical power for asymptotic connectivity in wireless networks, in: Stochastic Analysis, Control, Optimization and Applications (W.M. McEneany, G. Yin, Q. Zhang, eds.), Birkhauser, Boston, 1998, pp. 547-566.

[h] http://www.networks.digital.com/npb/html

[HCB] W.R. Heinzelman, A. Chandrakasan and H. Balakrishnan, Energy-efficient routing protocols for wireless microsensor networks, Proc. HICSS, Hawaii, January 2000, to appear.

[HKB] W.R. Heinzelman, J. Kulik and H. Balakrishnan, Adaptive protocols for information dissemination in wireless sensor networks, Proc. MOBICOM, 1999, Seattle, pp. 174-185.

[HL] T.C. Hou and V.O.K. Li, Transmission range control in multihop packet radio networks, IEEE Transactions on Communications, 34, 1, 1986, pp. 38-44.

[IETF] IETF Manet charter, http://www.ietf.org/html.charters/manet-charter.html .

[JM] D. Johnson, D. A. Maltz, Dynamic source routing in ad hoc wireless networks, in Mobile Computing (T. Imielinski and H. Korth, eds.), Kluwer Acad. Publ., 1996.

[K] E.D. Kaplan (ed.) Understanding GPS: Principles and Applications, Artech House, Boston, MA, 1996.

[KKKP] L.M. Kirousis, E. Kranakis, D. Krizanc, A. Pelc, Power consumption in packet radio networks, STACS, Lecture Notes in Computer Science Vol. 1200, 1997, pp. 363-374.

[KKP] J.M. Kahn, R.H. Katz, K.S.J. Pister, Next century challenges: Mobile networking for 'smart dust', Proc. MOBICOM, 1999, Seattle, pp. 271-278.

[KSU] E. Kranakis, H. Singh and J. Urrutia, Compass routing on geometric networks, Proc. 11th Canadian Conference on Computational Geometry, Vancouver, August, 1999.

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

[LL] C. R. Lin and J.S. Liu, QoS routing in ad hoc wireless networks, IEEE J. Selected Areas in Communications, 17, 8, Aug. 1999, pp. 1426-1438.

[LS] X. Lin and I. Stojmenovic, Geographic Distance Routing in Ad Hoc Wireless Networks, SITE, TR-98-10, University of Ottawa, December 1998.

[MC] J.P. Macker and M.S. Corson, Mobile ad hoc networking and the IETF, ACM Mobile Computing and Communications Review, 2, 1, 1998, pp. 9-14.

[MG] W. Mangione-Smith and P.S. Ghang, A low power medium access control protocol for portable multi-media systems, 3rd int. Workshop on Mobile Multimedia Communications, Sept. 1996.

[MS] P. Morin and I. Stojmenovic, Power aware routing algorithms with guaranteed delivery in wireless networks, in preparation.

[N] NAVSTAR GPS operations, http://tycho.usno.navy.mil/gpsinfo.html, and Iowa State University GPS page, http://www.cnde.iastate.edu/gps.html .

[NI] J.C. Navas and T. Imielinski, GeoCast - geographic addressing and routing, Proc. MOBICOM, 1997, pp. 66-76.

[NK] R. Nelson and L. Kleinrock, Tha spatial capacity of a slotted ALOHA multihop packet radio network with capture, IEEE Transactions on Communications, 32, 6, 1984, pp. 684-694.

[PL] K. Pahlavan and A. Levesque, Wireless information networks, Wiley Interscience, 1995.

[R] T.S. Rappaport, Wireless communications: Principles and Practice, Prentice Hall, 1996.

[RB] J.M. Rulnick and N. Bambos, Mobile power management for wireless communication networks, Wireless Networks, 3, 1997, pp. 3-14.

[RM] V. Rodoplu and T.H. Meng, Minimum energy mobile wireless networks, IEEE Journal on Selected Areas in Communications, Vol. 17, No. 8, August 1999, pp. 1333-1344.

[RS] S. Ramanathan and M. Steenstrup, A survey of routing techniques for mobile communication networks, Mobile Networks and Applications, 1, 2, 1996, pp. 89-104.

[S] I. Stojmenovic, Power-Aware routing in ad hoc wireless networks, SITE, TR-98-11, University of Ottawa, December 1998.

[SC] A.K. Salkintzis and C. Chamzas, An in-band power-saving protocol for mobile data networks, IEEE Transactions on Communications, 46, 9, 1998, pp. 1194-1205.

[SL1] I. Stojmenovic and X. Lin, GEDIR: Loop-free location based routing in wireless networks, IASTED Int. Conf. on Parallel and Distributed Computing and Systems PDCS'99, Nov. 3-6, 1999, Boston, MA, USA, pp. 1025-1028.

[SL2] I. Stojmenovic and Xu Lin, Power-aware routing in ad hoc wireless networks, 2000 IEEE International Parallel and Distributed Processing Symposium IPDPS2000, Mar. 2000, pp. 371-376, Cancun, Mexico.

[SR] S. Singh and C.S. Raghavendra, PAMAS – Power aware multi-access protocol with signaling for ad hoc networks, ACM Computer Communications Review, July 1998, pp. 5-26.

[SSL] A.R. Shahani, D.K. Schaeffer, and T.H. Lee, A 12mW wide dynamic range CMOS front-end for a portable GPS receiver, Proc. IEEE Int. Solid-State Circuits Conf., vol. 40, Feb. 1997, pp. 368-369.

[SWR] S. Singh, M. Woo, C.S. Raghavendra, Power-aware routing in mobile ad hoc networks, Proc. MOBICOM, 1998, pp. 181-190.

[SV] I. Stojmenovic and B. Vukojevic, A routing strategy and quorum based location update scheme for ad hoc wireless networks, Computer Science, SITE, University of Ottawa, TR-99-09, September 1999.

[SWR] S. Singh, M. Woo, C.S. Raghavendra, Power-aware routing in mobile ad hoc networks, Proc. MOBICOM, 1998, pp. 181-190.

[TK] H. Takagi and L. Kleinrock, Optimal transmission ranges for randomly distributed packet radio terminals, IEEE Transactions on Communications, 32, 3, 1984, pp. 246-257.