2011

RESOURCE ALLOCATION AND EFFICIENT ROUTING IN WIRELESS NETWORKS

Jianjun Yang
University of Kentucky, jjyang8888@gmail.com

Recommended Citation

Yang, Jianjun, "RESOURCE ALLOCATION AND EFFICIENT ROUTING IN WIRELESS NETWORKS" (2011). *University of Kentucky Doctoral Dissertations*. 833.
https://uknowledge.uky.edu/gradschool_diss/833

2011

RESOURCE ALLOCATION AND EFFICIENT ROUTING IN WIRELESS NETWORKS

Jianjun Yang
University of Kentucky, jjyang8888@gmail.com

Recommended Citation

Yang, Jianjun, "RESOURCE ALLOCATION AND EFFICIENT ROUTING IN WIRELESS NETWORKS" (2011). *Doctoral Dissertations.* Paper 833.
http://uknowledge.uky.edu/gradschool_diss/833

ABSTRACT OF DISSERTATION

Jianjun Yang

The Graduate School
University of Kentucky
2011

RESOURCE ALLOCATION AND EFFICIENT ROUTING IN WIRELESS NETWORKS

ABSTRACT OF DISSERTATION

A dissertation submitted in partial fulfillment of the
requirements for the degree of Doctor of Philosophy in the
College of Engineering
at the University of Kentucky
By

Jianjun Yang

Lexington, Kentucky

Director: Zongming Fei, Ph.D,

Associate Professor of Computer Science

Lexington, Kentucky

2011

ABSTRACT OF DISSERTATION

RESOURCE ALLOCATION AND EFFICIENT ROUTING IN WIRELESS NETWORKS

In wireless networks, devices (nodes) are connected by wireless links. An important issue is to set up high quality (high bandwidth) and efficient routing paths when one node wants to send packets to other nodes. Resource allocation is the foundation to guarantee high quality connections. In addition, it is critical to handle void areas in order to set up detour-free paths. Moreover, fast message broadcasting is essential in mobile wireless networks. Thus, my research includes dynamic channel allocation in wireless mesh networks, geographic routing in Ad Hoc networks, and message broadcasting in vehicular networks.

The quality of connections in a wireless mesh network can be improved by equipping mesh nodes with multi-radios capable of tuning to non-overlapping channels. The essential problem is how to allocate channels to these multi-radio nodes. We develop a new bipartite-graph based channel allocation algorithm, which can improve bandwidth utilization and lower the possibility of starvation. Geographic routing in Ad Hoc networks is scalable and normally loop-free. However, traditional routing protocols often result in long detour paths when holes exist. We propose a routing protocol-Intermediate Target based Geographic Routing (ITGR) to solve this problem. The novelty is that a single forwarding path can be used to reduce the lengths of many future routing paths. We also develop a protocol called Hole Detection and Adaptive Geographic Routing, which identifies the holes efficiently by comparing the length of a routing path with the Euclidean distance between a pair of nodes. We then set up the shortest path based on it. Vehicles play an important role in our daily life. During inter-vehicle communication, it is essential that emergency information can be broadcast to surrounding vehicles quickly. We devise an approach that can find the best re-broadcasting node and propagate the message as fast as possible.

Jianjun Yang

September 20, 2011

RESOURCE ALLOCATION AND EFFICIENT ROUTING IN WIRELESS NETWORKS

By

Jianjun Yang

Zongming Fei

Director of Dissertation

Raphael A. Finkel

Director of Graduate Studies

September 20, 2011

RULES FOR THE USE OF DISSERTATIONS

Unpublished dissertations submitted for the Doctor's degree and deposited in the University of Kentucky Library are as a rule open for inspection, but are to be used only with due regard to the rights of the authors. Bibliographical references may be noted, but quotations or summaries of parts may be published only with the permission of the author, and with the usual scholarly acknowledgements.

Extensive copying or publication of the dissertation in whole or in part also requires the consent of the Dean of the Graduate School of the University of Kentucky.

A library that borrows this dissertation for use by its patrons is expected to secure the signature of each user.

Name	Date

DISSERTATION

Jianjun Yang

The Graduate School

University of Kentucky

2011

RESOURCE ALLOCATION AND EFFICIENT ROUTING IN WIRELESS NETWORKS

DISSERTATION

A dissertation submitted in partial fulfillment of the
requirements for the degree of Doctor of Philosophy in the
College of Engineering
at the University of Kentucky

By

Jianjun Yang

Lexington, Kentucky

Director: Zongming Fei, Ph.D,

Associate Professor of Computer Science

Lexington, Kentucky

2011

ACKNOWLEDGEMENTS

It is a pleasure to thank those who made this thesis possible. I would never have been able to finish my dissertation without the guidance of my committee members, help from friends, and support from my family.

First of all, I would like to express my deepest gratitude to my advisor, Dr. Zongming Fei, for his excellent guidance, patience, caring, and leading me to do research in wireless networks. Dr. Fei has been a great mentor on every account, and his broad knowledge and constructive suggestions to this dissertation are sincerely appreciated.

I would like to thank the Director of Graduate Studies in the Department of Computer Science, Dr. Raphael Finkel and other faculty members of my Advisory Committee: Dr. Dakshnamoorthy Manivannan (Department of Computer Science), Dr. Jinze Liu (Department of Computer Science), and Dr. Robert Heath (Department of Electrical and Computer Engineering) for their helpful comments on my dissertation.

Also, I would like to thank Dr. Zongming Fei, Dr. Dakshnamoorthy Manivannan, Dr. Jinze Liu, Dr. Miroslaw Truszczynski and Dr. Debby Keen for their kind assistance with writing me recommendation letters and helping me in my job search.

Thanks also go to all members in the Laboratory for Advanced Networking, Dr. Libing Wu, Mr. Ping Yi, Ms. Shuhua Wang, and Ms. Jianning Liu for their kind help.

Finally I would like to thank my family members. I need to thank my parents for giving my life and their continuous support. I also need to thank my elder sister Ms. Jianmin Yang, who takes care of my parents in China and then I can perform

my research smoothly in the U.S.. Most important, I would like to thank my wife, Juan Guo, for her endless love and great patience during my study at the University of Kentucky.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

This thesis develops algorithms for channel allocation in Wireless Mesh Networks, geographic routing in Ad Hoc networks, and broadcasting in vehicular networks of wireless network environment.

Recent advances in communication technology and portable computing devices have resulted in the rapid development of wireless network systems. In wireless networks, nodes are equipped with wireless interfaces and remain connected to the network through wireless links. An essential issue in wireless networks is to provide high quality (high bandwidth) and efficient routing paths for nodes to send messages to other nodes. In current wireless mesh networks, nodes can be equipped with multiple network interfaces. Therefore, they can be connected with each other by different radios with different channels. In order to provide high quality connections, it is essential to allocate resources efficiently. In addition, there may be void areas (holes) in the network plane, so it is critical to handle the holes properly in order to set up the most efficient routing paths. Moreover, nodes in Ad Hoc networks may be mobile. In particular, in this thesis we focus on vehicles carrying Ad Hoc network devices acting as mobile nodes and investigate the broadcast problem in vehicular networks. My dissertation is focused on three problems in wireless networks, including channel allocation approaches for providing high quality connections, setting up efficient paths when there are void areas, and transmitting messages as fast as possible when moving nodes (vehicles) are present in mobile Ad Hoc networks.

1.1 Channel Allocation of Wireless Mesh Networks

Wireless mesh networks (WMNs) have emerged as one of the key technologies in the field of wireless communications. A wireless mesh network is a communication network made up of nodes organized in a mesh topology [1]. WMNs can connect diverse network nodes such as desktops, laptops, PDAs, PocketPCs and cell phones. Different networks, for instance, static Ad Hoc networks, mobile Ad Hoc networks, and sensor networks, can be connected into a WMN network. WMNs have many advantages such as low cost, easy network maintenance, robustness, and reliable service coverage. WMNs have inspired numerous applications due to their advantages over other wireless technologies. A typical wireless mesh network consists of mesh routers and mesh clients [1]. Networks, such as WiFi, 802.15, 802.16 and sensor networks, can be integrated into mesh networks through gateways and mesh routers. Mesh clients, either stationary or mobile, can link together, either by themselves or through connections with mesh routers.

WMNs are anticipated to significantly improve the performance of Ad Hoc networks, wireless local area networks (WLANs), wireless personal area networks (WPANs), and wireless metropolitan area networks (WMANs) [1]. An important measurement for the quality of connections in wireless networks is capacity (bandwidth). It is well-known that wireless interference severely limits network capacity in multi-hop settings. Traditionally, a WMN node was equipped with one IEEE radio [1] with one channel. As a result, this single-radio mesh network can only provide limited capacity

---

[1] IEEE stands for the Institute of Electrical and Electronics Engineers. Radio is the transmission of signals by modulation of electromagnetic waves with frequencies. IEEE radio is a radio defined by IEEE standard. For instance, IEEE defines the radio 802.11 with frequency bands 2.4, 3.6 and 5.18 GHz.

for clients. Fortunately, the physical characteristics of current IEEE radios make it possible for one node to be equipped with multiple network interfaces. Therefore, the node can simultaneously use multiple radios over non-overlapping channels to increase the overall capacity of a wireless mesh network. For example, a mesh node with two interface cards can be assigned one 802.11a channel with 5.18G frequency and one 802.11b channel with 2.4G frequency allowing this node to communicate with two other nodes over two different channels simultaneously, thus increasing the overall capacity of the network.

The method of assigning channels to mesh nodes is a major factor in increasing the efficiency of channel usage. To increase network bandwidths, we need to apply efficient algorithms for allocating existing channels. A basic requirement is to avoid interference because different links or users cannot use the same channel within their transmission range at the same time [6, 7]. Static channel allocation algorithms allocate fixed channel slices to each user. They can prevent interference, but these algorithms often result in poor utilization and spectrum holes [9]. To solve this problem, various dynamic channel assignment solutions have been proposed in the literature. Users can sense their available channels and utilize them opportunistically. This technology becomes possible because lower layer technical innovations equip nodes in wireless mesh networks with multi-radios and enable them to access different channels at different locations and times [5, 11, 12].

Dynamic allocation of channels can be categorized into two types [4]. One is the hierarchical access model, in which users are divided into primary users and secondary users. The channels are assigned to primary users first. Secondary users can use them only if the channels are released by primary users at certain time slots or the channels are free. The other is the dynamic exclusive use model, in which channels are allocated to a user for exclusive use at a certain time and may be re-allocated to a different user later. The objective of dynamic allocation is to improve the efficiency of using channels through flexible allocation strategies.

Most existing channel allocation approaches [16, 17, 18] are based on conflict-graph model, in which a vertex represents a user and an edge connects two users in case that they conflict. However, an edge can only represent the relationship of conflict. Because edges do not have weights, it is very difficult to use graph-theory based algorithms to achieve the objective of maximizing bandwidth utilization. Another problem with current solutions is that they focus on either increasing bandwidth utilization or decreasing starvation, but not both.

We propose a new bipartite-graph based model and design a channel allocation algorithm that considers both bandwidth utilization and starvation problem. Our solution is based on using augmenting path to find a matching in the bipartite-graph that can reduce the possibility of starvation and nearly maximize bandwidth utilization. Our simulation results demonstrate that our algorithm has less starvation ratio and better bandwidth utilization, compared to existing conflict-graph based algorithms.

1.2 Geographic Routing in Wireless Ad Hoc Networks

An Ad Hoc network consists of a collection of wireless communication nodes. Two nodes within their transmission range of each other can communicate directly. However, if a source node wants to send packets to a destination node outside its transmission range, it must depend on other nodes to relay the packets, because no fixed infrastructure exists in an Ad Hoc network. Many routing protocols (e.g., DSDV [47], AODV [49]) have been proposed for wireless Ad Hoc networks to find a path from the source to the destination. The main issue with these routing schemes is scalability because most of these schemes have to use flooding to find routing paths.

When location information for nodes is available (either through GPS or using virtual coordinates [19]), routing in Ad Hoc networks can be much more efficient.

Geographic routing exploits the location information and makes the routing in Ad Hoc networks scalable. The source node first acquires the location of the destination node it wants to communicate with, and then forwards the packet to its neighbor closest to the destination. This process is repeated until the packet reaches the destination. A path is found via a series of independent local decisions rather than flooding. However, geographic routing suffers from the so-called local minimum phenomenon, in which a packet may get stuck at a node that does not have a neighbor that is closer to the destination, even though a path exists from the source to the destination in the network. This typically happens when there is a void area (or hole) that has no active nodes. In wireless Ad Hoc networks, holes are caused by various reasons [25]. For instance, malicious nodes can jam communication forming jamming holes. If node signals are not long enough to collectively cover everywhere in the network plane, coverage holes may exist. Moreover, routing holes can be formed either due to voids in node deployment or because of nodes failure. Node failures can result from various reasons such as malfunctioning or battery depletion.

To deal with local minimum problem, Karp and Kung proposed the Greedy Perimeter Stateless Routing (GPSR) protocol, which guarantees the delivery of a packet if a path exists [26]. When a packet is stuck at a node because of the existence of a hole, the protocol routes the packet around the faces of the hole to get out of the local minimum. Several approaches have been proposed that originate from face routing. Although they can find the available routing paths, they often cause long detour paths.

We propose a new approach called Intermediate Target based Geographic Routing (ITGR) to avoid such long detour paths. The source node determines destination areas which are shaded by the holes based on previous forwarding experience. The novelty of the approach is that a single forwarding path can be used to determine an area that may cover many destination nodes. We design an efficient method for the source node to find out whether a destination node belongs to a shaded area. The source node then selects an intermediate node as the tentative target and greedily forwards packets to it to avoid the long detour path. Finally, the intermediate target forwards the packet to the destination by greedy routing. Simulations show that compared with GPSR, ITGR reduces the length of routing path by 15.4% and forwarding hops by 16.9% for all the paths in tested areas. Moreover, ITGR reduces the length of routing path by 70.5% and forwarding hops by 72.7% for the paths struck by holes.

To avoid long detour paths, recent work tries to detect holes and the nodes located on the hole's boundary in advance, and let these nodes advertise the hole information to some other nodes [31] [34]. In this way future routing paths can be adaptive in the presence of the hole.

We developed a Hole Detection and Adaptive Geographic Routing (HDAR) algorithm, which focuses on defining and detecting holes in Ad Hoc networks, representing detected holes, and building routes around the holes. The contributions of our algorithm are threefold. First, we come up with a heuristic algorithm to detect a hole quickly and easily. Our algorithm only needs a calculation with constant time complexity to identify a hole. Second, we provide a very concise format to represent a hole. Third, we develop a method for nodes located on the hole's boundary to announce hole information to the nodes in the vicinity of the hole. Simulations show that compared with GPSR, HDAR reduces the length of routing path by 12.4% and forwarding hops by 13.2% for all the paths in tested areas. In addition, the length of long detour paths around the hole can be reduced by 61.2%. The number of hops of these detour paths can be reduced by 64.6% compared with GPSR. Simulations also indicate that the computational complexity of HDAR is only 16.6% that of HAGR [45].

1.3 Broadcasting in Vehicular Networks

Vehicles play an important role in our daily life. Nowadays, more and more vehicles are equipped with wireless communication capabilities. Smart vehicles integrate environment-aware, route-planning, decision-making and drive-assistant technologies into the onboard system. These technologies utilize computers, sensors, communication, artificial intelligence and control technology to improve the safety and comfort level of vehicles [62]. Recent advances in wireless technologies have made Inter-Vehicle Communications possible which led to the new type of mobile networks known as Vehicular Ad Hoc Networks (VANETs) [55] [58]. VANETs have attracted considerable attention from both research community and automobile industry. Many automobile manufacturers have currently installed communication devices into their vehicles for purposes of security, convenience and comfort.

With the development of VANETs, many attractive applications have emerged [63]. The first application is Collision Avoidance. About 40 thousand deaths occur every year in the U.S. due to automobile accidents [57]. Effective communication between vehicles, and between vehicles and roads can save many lives and prevent injuries. Some of the worst traffic accidents involve many vehicles striking each other after a single accident suddenly halts traffic. In this application, once a vehicle reduces its speed significantly, it will broadcast its circumstance to its neighbor vehicles. The second application is Cooperative Driving. Examples of this are violation warning, curve warning, lane merging warning, etc. [31] [59]. These services may dramatically reduce accidents. Many accidents are due to lack of cooperation between drivers. Inter-Vehicle communications can prevent many of these accidents. The third application is Traffic Optimization. Traffic delays continue to increase, wasting a great deal of time for drivers and passengers. A significant reduction in traffic delay can be achieved through vehicular networks. Vehicles can serve as data collectors and transmit traffic condition information over vehicular networks. In addition, transportation agencies could utilize this information to actively relieve traffic congestion. In particular, each vehicle could detect the number of neighboring vehicles and their average speeds, and relay this information to other vehicles in order to prevent these other vehicles from moving towards the busy location. In some scenarios, the information could be relayed by vehicles moving in the opposite direction, allowing the information be propagated toward vehicles approaching the congested location faster. Vehicles could also collect data about weather, road surfaces, construction zones, highway rail intersections, and emergency vehicle signals, and relay this information to other vehicles [56] [65].

Message dissemination is an essential component of VANET applications. Researchers are developing methods for broadcasting messages to other vehicles. Designing effective methods for broadcasting in VANETs poses a number of challenges. In VANETs, the topology created by vehicles is usually very dynamic and significantly non-uniformly distributed [68]. Similar to MANETs, VANETs have no fixed infrastructure and instead rely on ordinary nodes to perform the routing of messages. However, VANETs behave in different ways than traditional MANETs. Vehicles move at much faster rate than nodes in MANETs [64]. It is also expected that communication between nodes that have never interacted before and will never interact again will be the norm. Thus VANETs are very different from MANETS. These characteristics imply that protocols designed for MANETs are not suitable for VANETs [63].

Mobility constraints and high dynamics are unique characteristics of VANETs. Velocities of vehicles are restricted due to speed limits, and traffic control mechanisms such as stop signs, traffic lights, and road conditions. Furthermore, VANETs encounter a major routing issue, i.e., the broadcast storm problem. The broadcast storm problem occurs when mobile nodes send messages by flooding, causing frequent link layer contention with other nearby broadcasting nodes, resulting in packet loss due to collisions. Therefore, it is extremely difficult to design a sound message broadcasting approach, especially one that can broadcast messages to other vehicles as fast as possible.

Current approaches employ distance-based mechanisms for broadcasting. In these schemes, the sender node tries to select the farthest node in the broadcasting direction and assign it to forward the message. These approaches may not select the best candidate for forwarding the message. Moreover, packets can be dropped because of network partitions. However, tracking mobility patterns of vehicles help vehicles carry packets to nodes in different partitions. If vehicles are moving between network partitions, then packets can be delivered even if the network is disconnected. This is the idea behind the concept of Delay Tolerant Networks (DTN) [69]. We develop a statistical filtering based DTN broadcasting algorithm. In this algorithm, each node maintains the acceleration and the variance of acceleration that can represent its historical mobility status. In addition, each node can predict its future mobility trend by considering the two parameters and its current velocity. In the broadcasting process, the sender disseminates a message to all its neighbors in the broadcasting direction. The sender selects the fastest candidate based on considering its neighbors' future mobility and designates this candidate to re-broadcast the message. Moreover, the strategy of DTN is incorporated in our algorithm. That is, when the network is disconnected, the re-broadcasting node can carry the message until it has at least one neighbor. The overhead of our algorithm is low because each node only needs to maintain two parameters. Our simulation results indicate that our approach can significantly decrease the end-to-end delay and improve the message delivery ratio, compared with existing approaches.

1.4 Organization of this Dissertation

The rest of this dissertation is organized as follows. In chapter 2, we review the related work regarding the topics of our research. In Chapter 3, we present our dynamic channel allocation algorithm for Wireless Mesh Networks. It is an effective approach that yielded near maximal bandwidth utilization and lower the possibility of starvation on channel assignment behavior. In Chapter 4, we present our Intermediate Target Based Geographic Routing approach for Ad Hoc networks. Our approach has the novel and powerful ingredient that a single forwarding path can be used to determine a shaded area that may cover many destination nodes. In Chapter 5, we come up with another approach for geographic routing in Ad Hoc networks, namely, Hole Detection and Adaptive Geographic Routing. This approach identifies the holes in the network efficiently by comparing the length of routing path with Euclidean distance between a certain pair of nodes. We then propose a concise representation of holes and present an effective routing scheme which sets up the shortest route. We present Statistical Filtering Based DTN Broadcasting Protocol for Vehicular Networks in Chapter 6. It is a new approach based on Kalman Filtering, in which the current message sender can select the best candidate that will re-broadcast the message to other vehicles as fast as possible. An important feature of this protocol is that choosing the next re-broadcasting node is based on the node's past velocity history and the prediction of its future velocity, allowing for messages to propagate faster. Finally, we conclude the dissertation and outline our future work in Chapter 7.

Chapter 2

Related Work

In this chapter, we review the research work related to our research.

2.1 Dynamic Channel Allocation in Wireless Mesh Networks

The strategies of dynamic channel allocation are critical to provide high quality connections over wireless nodes. Scholars have developed several approaches for channel allocation.

To use multiple channels with commodity hardware effectively, static channel allocations have been investigated [8, 14]. However, they cannot automatically change the allocation quotas when the network scenario changes.

In contrast to static allocation, another strategy is dynamic channel allocation. Most dynamic channel allocation mechanisms use heuristic algorithms to achieve increased bandwidth utilization. Zheng and Peng [17] proposed a greedy algorithm for dynamic channel allocation. Their algorithm picks the vertex with the highest bandwidth and assigns the channel to its associated user in each step. Then it cuts the edges that interfere with this user. It repeats these two steps until all the channels are allocated. This algorithm can increase bandwidth utilization without considering any other constraints. A problem with the algorithm is that it may cause starvation for some users.

Another approach [17] is to pick up the vertex with the highest label, which is defined as the bandwidth of the channel divided by the number of users interfering with each other over this channel. This approach tries to maximize the utilization and minimize the interference. However, it cannot allocate the channels to the maximal number of users.

Marina and Das [18] proposed a centralized greedy heuristic algorithm called CLICA for channel allocation. They use the node's degree of flexibility as a guide in determining the order of coloring decisions. Each node is associated with a priority. The algorithm starts to color the node (assign channels to the node) with the lowest priority and then color each of its adjacent nodes. It updates the graph until all the nodes are colored. This algorithm can achieve minimal interference and maintain a topology in a network. However, it does not consider the total bandwidth utilization.

Also related is Bhaskaran Raman's work on channel allocation [35]. Raman uses a bipartite graph to represent the traffic fraction between a pair of nodes. The traffic fraction from a given node to another is defined as f, the traffic fraction in the opposite direction is $1-f$. The objective of his algorithm is to minimize the mismatch, that is, to minimize the difference between the desired match fraction DF and the achieved fraction AF. Graph coloring method is used to achieve the objective. Raman also uses bipartite graph to mainly represent the relationship between traffic fractions in opposite directions, while we use augmenting path algorithms based on the bipartite graph for channel assignment.

2.2 Geographic Routing in Wireless Ad Hoc Networks

Many geographic routing protocols have been developed for MANETs. In early protocols, each intermediate node in the network forwards packets to its neighbor closest to the destination, until the destination is reached. Packets are simply dropped when greedy forwarding causes them to end up at a local minimum node.

To solve local minimum problem, the geometric face routing algorithm (called

Compass Routing) [57] was proposed. Compass Routing guarantees packet delivery in most, (but not all) networks. Several practical algorithms, which are variations of face routing, have been developed. By combining greedy and face routing, Karp and Kung proposed the Greedy Perimeter Stateless Routing (GPSR) algorithm [26]. It consists of a greedy forwarding mode and a perimeter forwarding mode. The perimeter forwarding mode is applied in the regions where the greedy forwarding fails. An enhanced algorithm, called Adaptive Face Routing (AFR) [40], uses an ellipse to restrict the search area during routing so that in the worst case, the total routing cost is no worse than a constant factor of the cost for the optimal route. To our knowledge, the latest addition to the family of face routing protocols is GPVFR, which improves routing efficiency by exploiting local face information [23].

To support geographic routing better in large wireless networks, several schemes have been proposed to maintain geographic information on planar faces [27]. In recent work [31, 32, 45], methods of finding void areas in advance were explored and used. A node keeps the coordinates of key nodes as well as the locations of its neighbors. The forwarding nodes will use the information to avoid approaching the holes.

Sundar, Sanjay and Piyush [61] proposed a geographic routing algorithm that typically achieves high throughput. This algorithm determines a rectangle that is around a hole. Then the routing path goes around the rectangle. Their analysis of the protocol shows that their algorithm can reach near-optimal throughput over random planar networks with an arbitrary number of routing holes of varying sizes.

Also related is GLR, a novel geographic routing scheme for large MANETs [36]. In their algorithm, once a source node sends packets to a destination node and meets a hole, the source node saves the location of the landmark node to its local cache. If any packet is to be forwarded to the same destination, the source node will forward the packet through the landmark. So each entry in the cache can only be used for a single destination node. In contrast, our approach learns from previous experience and generalizes it to cover destination nodes in an area. The number of nodes that will benefit from one cache entry can be orders of magnitude greater. We also designed a simple way to represent the area and an efficient algorithm to decide whether a destination node is in the area.

A recent technology can detect holes in a network environment in advance, then the nodes located on the hole advertise the hole information to other nodes. This information benefits nodes who receive it for their future routing. Qing gave a mathematical definition of hole [31]. He defined a hole to be a simple region enclosed by a polygon cycle which contains all the nodes where a local minimum can appear. He introduced the "get stuck" concept and proposed a hole detection mechanism that considers a node to be on the boundary of a hole once a packet following geographic greedy forwarding gets stuck at this node. Also related is HAGR [45]. HAGR investigated the nodes incident to a close loop in a geographical graph. For a vertex u, if the angle between two adjacent edges with respect to this vertex is larger than an threshold value, then vertex u considers itself to be located on the boundary of a potential hole. To further determine if it is located on the boundary of a hole, u calculates the diameter of the potential hole. It locates the bisector that equally splits the angle and uses it as a reference line. Then node u finds out the leftmost node and the rightmost node furthest from the bisector. The distance between the leftmost node and the rightmost node is the diameter of the hole. If the diameter is greater than the diameter threshold and the angle is bigger than the angle threshold, u is regarded as sitting on a hole. Once a node detects itself to be on a hole, it advertises the hole information to its neighbors. Upon receiving the hole information, its neighbors recalculate the angle and diameter based on their locations. If both of the angle and diameter are bigger than a neighboring node thresholds, then the neighbor considers itself to be on a hole and it continues to advertises the hole information, otherwise it stops advertisement. Based on the hole detection steps, HAGR divides the network plane into three regions, and the nodes in different regions conduct different forwarding strategies. HAGR can find holes of most scenarios. However, the hole detecting approach is time-consuming because a node has to calculate the values of two metrics. And hole advertisement is expensive because once a node receives hole information, it has to recalculate two values and compare them with their corresponding thresholds. In addition, the diameter threshold is an absolute value and has to be adjusted according to the nodes' transmission range or the network deployment, otherwise false negatives or false positives may occur. Moreover, the forwarding strategies are too complicated.

2.3 Message Broadcasting in Vehicular Networks

This section reviews the work on message broadcast protocols of VANETs and routing protocols for DTN. Applications developed for VANETs have a very specific and clear goal of providing an intelligent and safe transportation system. Emergency warning for public safety is one of many applications that is highly time-critical and requires a more intelligent broadcast mechanism than just blind flooding.

2.3.1 Broadcasting in VANETs

In [48], the authors proposed a spatially aware packet routing algorithm to disseminate the message in VANETs. This algorithm predicts the permanent holes in the topology. Then, geographic forwarding is conducted when messages are disseminated to other nodes. This approach guarantees messages will be broadcast to other nodes. However, it has high overhead because the void areas need to be detected ahead. In addition, the locations of nodes change frequently in a VANET.

Bai and Helmy [66] studied the impact of nodes' mobility on the topology of MANETs. However, their contribution is only part of a larger framework aimed although they explain the performance of routing protocols in mobile Ad Hoc networks, and thus suffers from a low level of detail. Also, their work only considers completely random motion representations and very approximated vehicular mobility models, inducing results hardly applicable to real-world vehicular networks.

A. Amoroso proposed a broadcasting algorithm for VANETs called FROV [50]. This algorithm is suitable in networks where vehicles have heterogeneous transmission ranges. In addition, the transmission range of a vehicle can vary while moving, due to changes in environmental conditions, such as humidity, rain, snow and fog. Moreover, topological conditions, such as tunnels, sharp curves, and surrounding trees or buildings, can further influence transmission ranges. The main idea of FROV is that the node that is selected to re-broadcast a message is the one whose re-transmission spans farther than other neighbors. To accomplish this task, FROV considers both the position and the transmission range of neighbors in the direction of the broadcast. This algorithm can handle scenarios with extreme weather or poor conditions, but it does not consider the mobility status of the vehicles, which is essential in VANETs.

A multi-hop broadcast protocol for inter-vehicle communication was proposed in [52]. This protocol divides the road into segments, then it chooses the vehicle in the farthest non-empty segment for re-broadcasting. When there is an intersection in the path of the message dissemination, new directional broadcasts are initiated by the repeaters located at the intersections. This protocol only considers the longest distance factor when selecting the best re-broadcasting node.

An approach that considers the velocities of the vehicles was proposed in [53]. In this algorithm, vehicles are divided into several clusters. Each cluster head maintains the moving status and density of the vehicles in its cluster. This approach takes mobility into consideration, but it is very difficult to maintain clusters in VANETs.

2.3.2 DTN Routing in VANETs

Several researchers have proposed DTN routing protocols for VANETs. Musolesi developed Context Aware Routing (CAR) [71]. It integrates synchronous and asynchronous mechanisms for message delivery. In this scheme, a synchronous message delivery mechanism is determined by a contemporaneous path between the current node and the destination. On the other side, an asynchronous message delivery mechanism does not have such a path. A node relays to another node with the highest probability of reaching the destination by the evaluation and prediction of the context information on the asynchronous message delivery. In CAR approach, Musolesi uses DSDV of traditional Ad Hoc routing and introduces prediction to reduce the overhead for dissemination of routing entries. Musolesi also provides one more framework [71] of utilizing the contextual information with dynamic-weight consideration geared towards sensor networks and prediction geared towards proactive routing.

GeOpps [72] is a delay tolerant routing algorithm that exploits the availability of information from a navigation system. The navigation system includes a GPS device, maps, and the function to calculate a potential route from current position to a given destination. Each node with GPS communicates with one another and obtains information to perform efficient and accurate route computation. When a vehicle wants to deliver a data packet, it broadcasts the destination. The one-hop neighbors of the packet holder will calculate the "Nearest Point." The "Nearest Point" is the location that is on the path which is geographically closest to the destination because every vehicle using navigation system has a suggested path.

A limitation of the GeOpps approach is that the scheme assumes all vehicles have a navigation system and the navigation system provides the same transmission format and content. The real world setting often differs from this assumption. For instance, GeOpps does not use heterogeneous information from devices other than the navigation system and misses opportunities of finding a better forwarder.

Nain et al. proposed the Mobile Relay Protocol (MRP) [73]. MRP uses a relay based approach conjunction with a traditional Ad Hoc routing protocol. A node uses a traditional routing protocol until a route to the destination is unobtainable. The node then performs controlled local broadcast to its immediate neighbors. All nodes that receive the broadcast store the packet and enter into the relaying mode. Such nodes carry the packet until their buffer is full. When that happens, the relay-nodes choose to relay the packet to a single random neighbor. In this mechanism, nodes constantly seek the best neighbor for delivering packets to the destination, until the buffer is full or until the relay node meets the destination which results in increased end-to-end delay.

Chapter 3

Dynamic Channel Allocation in Wireless Mesh Networks

In this chapter, we present our dynamic channel allocation algorithms. They are essential foundations to realize high quality (high bandwidth) connections over wireless nodes.

3.1 Problem Formulation

In a mesh network, the routers are relatively stationary. However, mesh clients, such as laptops and PDAs, can be mobile [3]. Fig. 3.1 gives an example of a wireless mesh network, in which the mesh routers are equipped with multiple IEEE 802 family radios. The routers need not be equipped with the same number of radios nor do they need to use identical types of radios. The types of radios and the number of channels depend on the number and physical parameters of the router's interfaces. At least one router in the mesh is designated as the gateway, which provides connectivity to an external network such as the Internet. The Channel Assignment Server (CAS), which is co-located with the gateway (Fig. 3.1), performs the task of channel assignment. Access points are co-located with mesh routers. Dotted lines in Fig. 3.1 illustrate that there could be multiple possible channels to be assigned to a node.

Available channels differ in bandwidth and transmission range. A channel might be available to multiple users, but it can only be allocated to one of them at any given

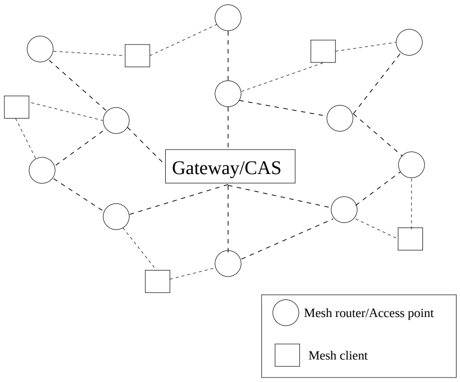

Figure 3.1: An example of wireless mesh network

moment if they are within transmission range of each other, otherwise interference will occur. At any given time, users are competing for available channels. Some objectives of channel assignment scheme are: (i) the number of users with allocated channels is maximal; and/or (ii) the sum of the allocated channel bandwidth is maximal.

A Bipartite graph [24] $G = (V, E)$ is used to model interference and available bandwidth among different users. In this model, the vertex set is composed of elements from two subsets, the user set U and the channel set C. That is, $V = U \cup C$ and $U \cap C = \emptyset$. Edge $e \in E$ is in the form of (u, c) where $u \in U$ and $c \in C$. Edge $e = (u, c)$ means that channel c is available to user u. For each user vertex $u \in U$, there is at least one edge connecting it. Otherwise, the node can be removed from the graph. The same is true for channel vertices. A weight function $W : E \longrightarrow R^+$ over the edge set E is further defined. The weight $W(e)$ of edge $e = (u, c) \in E$ is the bandwidth that user u can get if it uses channel c.

Generally, if multiple user vertices connect with the same channel vertex, they will conflict with each other. However, this depends on how the set of channel vertices is defined. Fig. 3.2 illustrates a case where users u_1, u_2, u_3 and u_4 can all possibly use the same channel c_1. Suppose that u_1 and u_2 are close to each other and u_3 and u_4 are close to each other, but u_1, u_2 are far away from u_3, u_4. So u_1 will interfere with u_2 while u_3 will interfere with u_4, but u_1, u_2 will not interfere with u_3, u_4 with

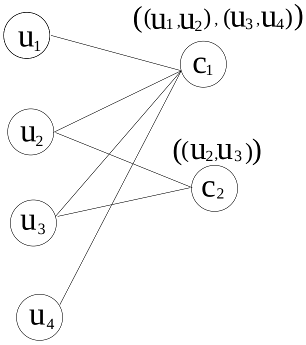

Figure 3.2: An initial bipartite graph to represent conflicts

regard to channel c_1. In this case, c_1 can be split into two channels, $c_{1,1}$ and $c_{1,2}$, to represent the channel in different locations. We then let u_1 and u_2 connect with $c_{1,1}$ and u_3 and u_4 connect with $c_{1,2}$, as shown in Fig. 3.3. With this simplification, we can make sure that if two user vertices connect with the same channel vertex, they will interfere with each other. For the rest of this section, we will assume that this split has already been done.

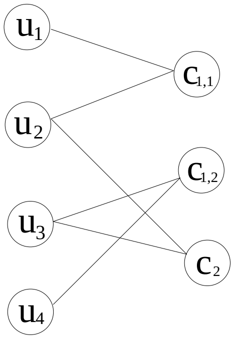

Figure 3.3: A simplified bipartite graph to represent conflicts

The channel assignment problem is to find a subgraph $G' = (V, E')$, where $E' \subseteq E$, such that for any $c \in C$, there exists only one $u \in U$, such that $(u, c) \in E'$. However, for a $u \in U$, it is fine that there are multiple edges connecting with it. The maximal bandwidth utilization problem can be defined as finding $G' = (V, E')$ such that $\sum_{e \in E'} W(e)$ is maximized.

The goal of minimizing starvation is to have as many users as possible being allocated with channels. We can define $U' = \{u \in U : \exists c \in C, (u, c) \in E'\}$. The minimal starvation allocation problem can be defined as finding $G' = (V, E')$, such that $|U'|$ is maximized.

3.2 Channel Assignment Algorithms

A bipartite graph matching algorithm is used to solve the channel assignment problem. The solution to the matching problem allocates at most one channel for one user at a given time. Our approach repeats the process multiple times so that a user can be allocated with multiple channels. We start with a simple solution of using maximal cardinality matching to find the solution to the channel assignment problem for minimizing starvation. Based on that, we propose a solution to the maximal weight matching problem, which will be used as the basis for the final dynamic channel assignment algorithm that lowers starvation and nearly maximizes bandwidth usage.

3.2.1 Maximal Cardinality Matching Problem

For a given bipartite graph $G = (V, E)$, a matching M is a subset of E such that no two edges in M have common vertex. A maximum cardinality matching is a matching that maximizes $|M|$. A naive algorithm to achieve maximum matching is to find out all the matching and select the matching with maximal edges. Its time complexity is exponential. We will present an alternative approach based on augmenting path [24].

Suppose M is a matching of graph G. The vertices adjacent to the edges in M are said to be matched. If P is a path connecting two unmatched vertices in G and the edges belonging to M and not belonging to M appear in P alternately, then P is an augmenting path based on M.

The augmenting path from v_i to v_j has three characteristics:

1. The number of edges in the augmenting path is an odd number.

Algorithm 1 *MaxCardMatching(G)*

```
1: $M \leftarrow \emptyset$,$E_{match} \leftarrow \emptyset$, $E_{unmatch} \leftarrow E$, $t \leftarrow number\ of\ unmatched\ edges$
2: $AugExist \leftarrow True$
3: while AugExist do
4:     $P \leftarrow \emptyset$
5:     for $e_i$=$e_1$,$e_2$,...,$e_t$ in $E_{unmatch}$ do
6:         $Success \leftarrow True$, $u_i \leftarrow\ left\ vertex\ of\ e_i$
7:         while Success do
8:             if not exist $c_j$ satisfying $(u_i, c_j) \in E_{unmatch}$ then
9:                 Success ← False
10:            else
11:                Find $c_j$ satisfying $(u_i, c_j) \in E_{unmatch}$,$P \leftarrow P \cup (u_i, c_j)$
12:                Find $u_k$ satisfying $(c_j, u_k) \in E_{match}$,$P \leftarrow P \cup (c_j, u_k)$,$u_i \leftarrow u_k$
13:            end if
14:        end while
15:    end for
16:    $M_{Pre} \leftarrow M$,$M \leftarrow M \oplus P$
17:    Update $E_{match}$, $E_{unmatch}$ and $t$ according to the edges in $M$
18:    if $M_{Pre} == M$ then
19:        AugExist← False
20:    end if
21: end while
22: Output M
```

2. Neither v_i nor v_j is incident to any edges in M.

3. A larger matching M' can be obtained by M and an augmenting path P based on M. Let $M' = M \oplus P$. That is, the larger matching M' includes the edges that either belong to M or belong to P but do not belong to both M and P.

The solution to the maximal cardinality matching problem is given in Algorithm 1. Initially, it sets the largest edge matching as Ø, and divides the edges as matched and unmatched edges. e_i represents an edge in the algorithm. Then from line 3 to line 14, it tries to find an augmenting path based on M to increase the matching cardinality. The method to augment matching edges is to let $M \leftarrow M \oplus P$. If no augmenting path can be found, which is determined by line 15, the algorithm stops and outputs the maximal cardinality matching.

For example, in Fig. 3.4 we show a matching M represented by solid lines in graph

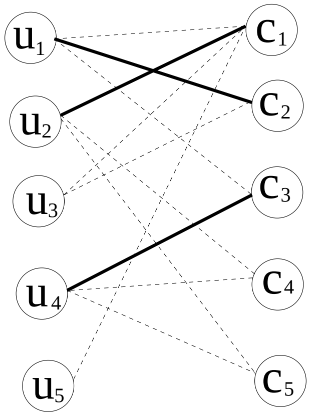

Figure 3.4: An example of maximal cardinality matching

G. An augmenting path P based on M found by algorithm 1 is shown in Fig. 3.5. By combining M and P, we obtain a larger matching $M^{'} = M \oplus P$ shown in Fig. 3.6.

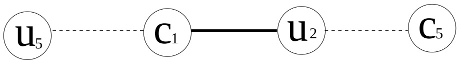

Figure 3.5: An example of augmenting path

Based on the tentative matching, we will next design an algorithm to construct an augmenting path to find a matching with maximal number of edges.

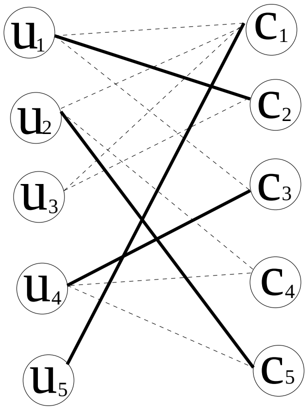

Figure 3.6: The process to generate a larger matching

3.2.2 Maximal Weight Matching Problem

To compute maximal weight matching, we would like to use an augmenting path approach to compute larger matching incrementally, eventually computing a maximal matching. The problem is how to find an augmenting path P based on a tentative matching M with k edges, and compute $M^{'} = M \oplus P$ a matching with $k+1$ edges. A brute-force algorithm is to list all the possible matching with $k+1$ edges and select the maximal one. Its time complexity is exponential. We will make use of a modified Bellman-Ford algorithm to solve this problem in Algorithm 2.

Initially, our algorithm sets the largest weight matching as $\emptyset$ and divides the edges as matched and unmatched. Then from line 2 to line 13, our algorithm tries to find an augmenting path based on M to enlarge the matching weight by the Bellman-Ford shortest path algorithm. The weight of an unmatched edge is negative, and the Bellman-Ford algorithm permits negative distance paths to exist. Hence once the path is shorter, the weight is bigger. The method to augment matching edges is still to let $M \leftarrow M \oplus P$. If no augmenting path can be found, the algorithm stops and outputs the maximal weight matching.

Fig. 3.7 illustrates the maximal weight matching with two edges. The weight of each edge is shown as a number beside it. The dark vertices denote matched vertices and the solid lines denote matched edges, while dashed lines denote unmatched edges.

In Fig. 3.8, a tentative matching M has two edges. There are multiple options for 3 edges matching. By algorithm 2, we found that augmenting path P is (u_4,c_2,u_2,c_3,u_3,c_5) because it is the shortest path with length -3. So the sum weight gained from this path is maximal. Let $M^{'} = M \oplus P$, then $M^{'}$ is ((u_4,c_2),(u_2,c_3),(u_3,c_5)). It is the maximal weight matching with 3 edges.

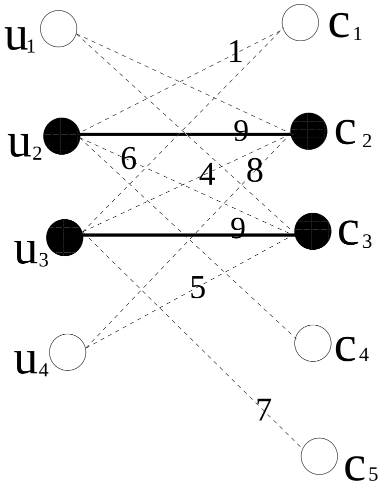

Figure 3.7: An example of maximal weight matching with 2 edges

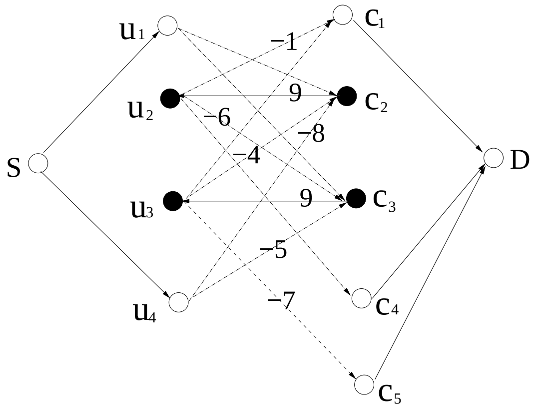

Figure 3.8: An example of modified graph to achieve maximal weight matching

Algorithm 2 *MaxWeightMatching(G)*

```
$M \leftarrow \emptyset$,$E_{match} \leftarrow \emptyset$,$E_{unmatch} \leftarrow E$,$k \leftarrow 0$, $AugExist \leftarrow True$
while AugExist do
    for any edge e$\in E_{match}$ do
        Let its direction be from S to U
    end for
    for any edge e$\in E_{unmatch}$ do
        Let its direction be from U to S,$W_e = -|W_e|$
    end for
    Add vertex S and D
    for any left vertex $u_i$ $in$ $E_{unmatch}$ do
        Add edge(S,$u_i$)
    end for
    for any right vertex $c_j$ in $E_{unmatch}$ do
        Add edge($c_j$,D)
    end for
    From S to D, find the shortest path $P$ by Bellman-Ford algorithm
    $M_{Pre} \leftarrow M$, $M \leftarrow M \oplus P$
    Update $E_{match}, E_{unmatch}$ and $k$ according to the edges in $M$
    if $M_{Pre} == M$ then
        AugExist ← False
    end if
end while
Output $M$
```

3.2.3 Dynamic Channel Assignment Algorithm

The final dynamic channel assignment algorithm is given in Algorithm 3. Its functionality is to allocate channels to as many users as possible and achieve maximal sum bandwidth as well.

In the first step, it creates graph $G^{'}$ which has the same nodes as G but no edges (i.e. no matching). $G^{'}$ is to accept the allocated channels each time. Lines 4 to 10 use a greedy approach to add matching to G'. It calls the $MaxWeightMatching$ algorithm multiple times to achieve the maximal cardinality and the maximal weight matching. It assigns at most one channel for each user each time. In every loop, once the algorithm computes such matching by $MaxWeightMatching$, it adds the matching to $G^{'}$ and deletes the interference edges in G. It repeats the operation until

Algorithm 3 *FinalMatching(G)*

```
let $G^{'} = (V^{'}, E^{'})$
$V^{'} \leftarrow V$
$E^{'} \leftarrow \emptyset$
while $E \neq \emptyset$ do
    M=MaxWeightMatching(G)
    for any edge $e_i \in$ M do
        $c_i \leftarrow e_i$'s right vertex
        delete $c_i$ from G
        delete edge adjacent to $c_i$ from G
    end for
    $G^{'} = G^{'} \cup M$
end while
Output $G^{'}$
```

G has no edge. The matching in $G^{'}$ represents the maximal matching, in which each node may be allocated with multiple channels.

Here is an example to illustrate how $FinalMatching$ works. Fig. 3.9 shows the original status of the channels and nodes. The dotted lines denote the possible availability relationships. Fig. 3.10 shows the channels that are assigned to the nodes after the while loop in Algorithm 4 goes through once. At the same time, the channels c_1, c_3 and c_6 and their adjacent edges are removed. Then the residual possible available channels for the nodes are shown as Fig. 3.11. The while loop of the algorithm will continue until all the channels are assigned.

In standard algorithm of creating an augmenting path in Bipartite Graph, the process is similar to BFS or DFS. Hence, the time complexity to create an augmenting path is $O(E)$ [24]. We use Bellman-Ford algorithm to find an augmenting path. The time complexity of Bellman-Ford algorithm is $O(VE)$. Therefore, the time complexity to figure out an augmenting path in our algorithm is $O(VE^2)$. Our $FinalMatching$ algorithm repeats at most $\mid E \mid$ times. Thus, our algorithm is a polynomial time complexity algorithm.

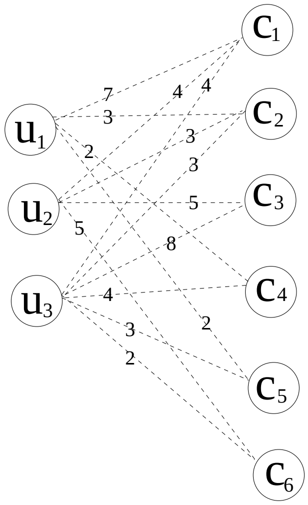

Figure 3.9: Illustration of possible available channels for nodes

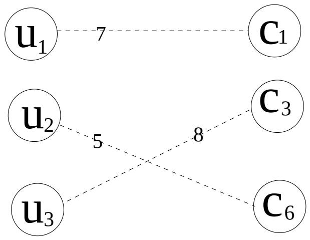

Figure 3.10: An example of channels allocated to nodes

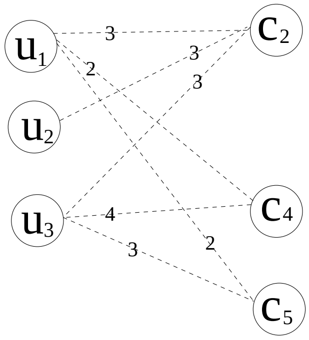

Figure 3.11: The residual possible available channels for nodes

3.3 Experimental Results

Our performance evaluation is conducted in a noiseless immobile radio network environment, where the nodes are distributed in a given area and each may have a different transmission range and bandwidth. Since what we are interested in is to compare our results with the outputs generated by other allocation approaches, we convert this network to a weight graph $G = (V, E)$, where the weight represents the bandwidth. We test both sparse and relatively dense network scenarios. We set the number of users to be 25 and 50, respectively.

For each case with a certain number of users, we let the number of channels vary from 25 to 50 with increment 5. We set the probability that an edge exists between any pair of user node and channel node to be 0.2 and the edge weight is uniformly distributed from 1 to 9. For each configuration, we generate 10 graphs and conduct the experiment 10 times based on the graphs. We calculate the average value from the experiments as the result for each configuration.

Two metrics are used to evaluate the performance. One is the sum of the allocation of bandwidth by all users. The other is the ratio of the number of users who are allocated with at least one channel over the number of users who are competing for the spectrum pool.

The bipartite-graph based solution is compared with three other approaches. The first one is NMSB (Non-collaborative-Max-Sum-Bandwidth). In each step, this approach picks the vertex with the highest bandwidth and assigns it to its associated user. Then the algorithm removes the edges that interfere with this user until all the channels are allocated. The second one is CMSB (Collaborative-Max-Sum-Bandwidth). This approach picks up the vertex with the highest label, defined as the bandwidth of a channel divided by the number of users interfering with each other with regard to this channel. The process is repeated until all channels have been allocated. The third approach is MINSTARVE, which tries to allocate the channels to the users who have not been allocated before. In this approach, each user has a priority. The user's priority is decreased by one if the user is allocated with a channel. The algorithm takes care of users by their priorities. Thus, it can assign channels to those who are in starvation status first.

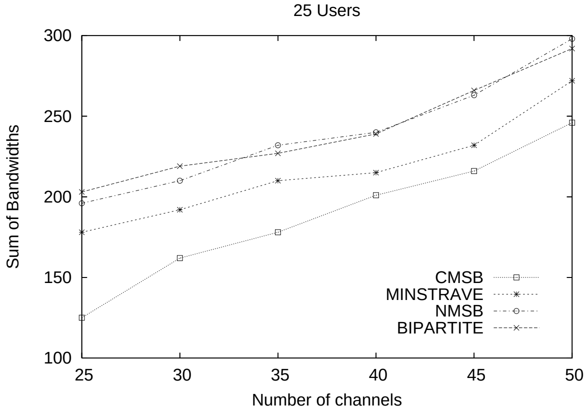

Figure 3.12: Performance evaluation of total bandwidth based on 25 users

In Fig. 3.12, the sum of bandwidths of these approaches is evaluated. The number of channels changes from 25 to 50. The results show that our approach can achieve near optimal sum bandwidth, similar to NMSB. Both NMSB and our approach are about 10% to 25% higher than the other two approaches.

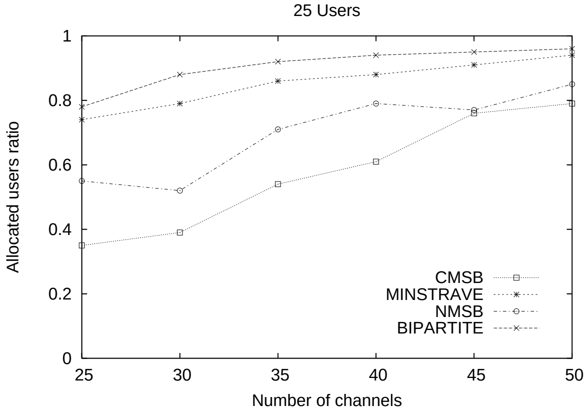

Figure 3.13: Performance evaluation of allocation ratios based on 25 users

The allocation ratio of the four approaches is reported in Fig. 3.13. It shows that our algorithm can reach near 90% allocation ratio. MINSTARVE is close to 80%, while the other two range from 40% to 70%. It illustrates that our mechanism can avoid starvation better.

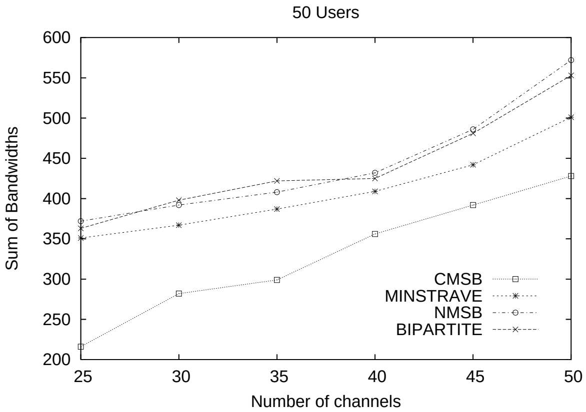

Figure 3.14: Performance evaluation of total bandwidth based on 50 users

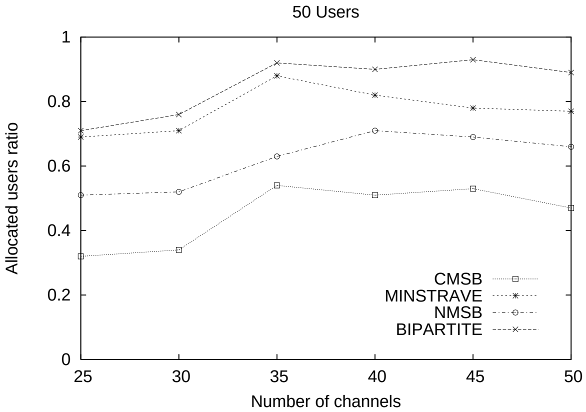

Figure 3.15: Performance evaluation of allocation ratios based on 50 users

The sum of bandwidths and allocation ratio of relatively dense networks are then evaluated and get the results are shown in Fig. 3.14 and Fig. 3.15 respectively. They illustrate that like sparse network, both NMSB and our approach can achieve high utilization and the bipartite-graph based algorithm can reach near 95% allocation ratio. The high percentage of allocation radio indicates that our bipartite-graph based algorithm can lower the possibility of starvation effectively.

Chapter 4

Intermediate Target Based Geographic Routing in Ad Hoc Networks

In this chapter, we present a geographic routing protocol that can set up efficient routing paths in the presence of holes in the network. A hole is an area without active nodes. It is also called a void area.

4.1 Shaded Area Detection

A method to find detour-free routing paths in geographic routing is discussed in this chapter. We assume that all the nodes lie in a two dimensional plane. Every node knows its location, either by being equipped with a GPS receiver or by virtual coordinates calculated through predefined algorithms. Each node also knows its neighbors' locations. When a node intends to send packets to a destination node, it knows the destination node's location. Every node has the same transmission range. All the nodes are static. That is, mobility is not considered.

4.1.1 Approach Outline

We start this section with a simple example to illustrate the basic idea behind our approach. In fig. 4.1, S is a source node and D_1, D_2 and D_3 are three different destination nodes. When S wants to send packets to D_1, it can find an efficient path using greedy forwarding. However, when S wants to send a packet to D_2 using greedy forwarding. Greedy forwarding fails when the packet reaches P due to the existence of the void area. So P is called a *local minimum node*. Algorithm such as the one in [26] lets P change from greedy mode to perimeter routing mode, and forward the packet along a detour path until it arrives at node B, where the forwarding mode is changed from perimeter routing mode to greedy forwarding. Node B is called a *landmark node*. After node B, the packet can be forwarded to destination D_2 by greedy forwarding. Due to the existence of the hole, the detour path taken by the packet can be long.

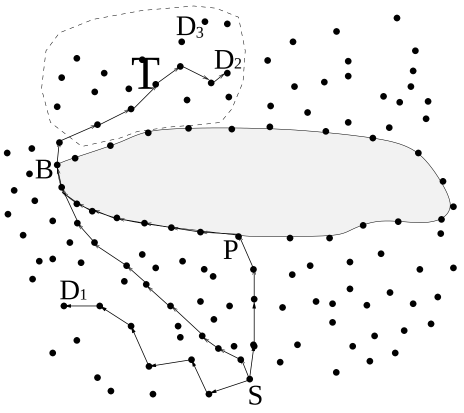

Figure 4.1: Comparison of greedy path and detour path

It can be helpful for the source node in the future if we can let either destination D_2 or landmark node B inform source S that such a detour occurred. Such information can be placed in the cache of the source node S in the form of $< D_2, B >$, meaning that if the destination is D_2, forward through intermediate node B. After that, if S later needs to send packets to D_2, it can send them to B first (using B as an intermediate target) by greedy forwarding. The path will be from S to B and then to D_2, instead of from S to P, to B, and then to D_2. This new path can be much shorter and may be the best path to get to D_2 from S. The significance of this technique depends upon the likelihood that S needs to send packets to D_2 again.

Now suppose that S needs to send a packet to D_3. Most likely, it will be forwarded to P by greedy forwarding, then go through the detour using perimeter routing to B, and finally reach D_3. The problem we are interested in is whether the detour information cached for destination D_2 can be used by S for forwarding packets to D_3. In other words, can we use the cached information about the void area for determining route to multiple destination nodes?

Our goal is to determine an area T shown as a shaded area in the figure such that for any destination node $D \in T$, source node S can use B as an intermediate target and prevent perimeter forwarding. The challenge is to find a simple representation of area T and an efficient algorithm to determine whether a target node D is in the shaded area.

4.1.2 Shaded Areas

In order for S to calculate the shaded area, the locations of the local minimum node P and landmark node B must be learned. When a packet arrives at a node in perimeter mode, this node will determine whether it is a landmark node by checking whether it should change the forwarding mode to greedy. If it is a landmark node, it will inform the source node of its own location (B) and the location of the local minimum node P (recorded in the packet).

S learns the locations of B and P when S first sends packets to destination D in case there is a hole that causes a detour path. The shape of a hole can be a convex polygon (Fig. 4.2) or a concave polygon (Fig. 4.3, Fig. 4.4, and Fig. 4.5). Additionally, from S to B, there can be a greedy path (Fig. 4.2 and Fig. 4.4) or non-greedy path (Fig. 4.3 and Fig. 4.5). Although the shapes of holes can differ, and a greedy path may exist or may not exist, we can use a uniform approach to find the potential shaded destination area. However, we need to use different processes to set up a routing path depending on whether there exist a greedy path from S to B or not.

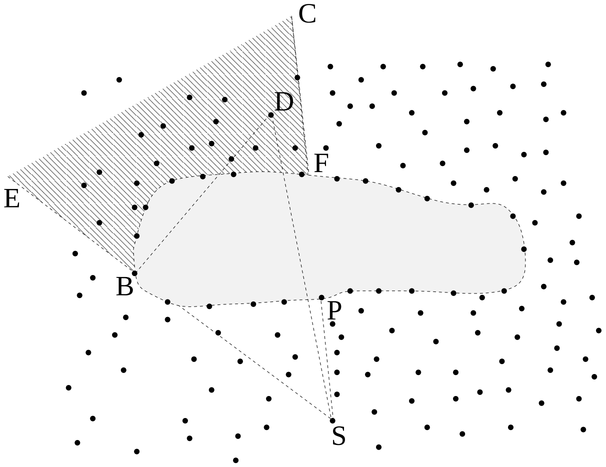

Figure 4.2: An example illustrating shaded area based on convex hole and greedy path

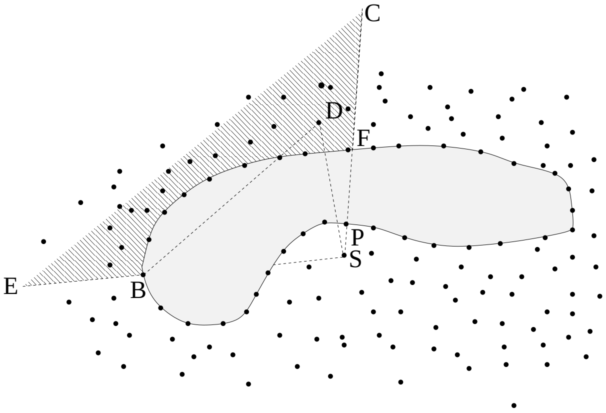

Figure 4.3: An example illustrating shaded area based on concave hole and non-greedy path

The method to find a shaded area can be summarized as follows (Fig. 4.6). S is connected with P. Suppose SP intersect with the hole at another point F. Ray SF is further extended to some point C. S is connected with B and is extended to some point E. Then the area semi-enclosed by EB, arc BF and FC is the shaded destination area T. So instead of cache entry (D_2, B), we have an entry (T, B) at node S. Hence, if S needs to send packets to any destination node D in T, the destination

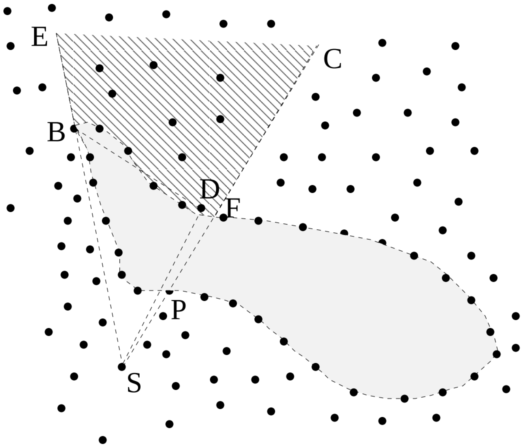

Figure 4.4: An example illustrating shaded area based on concave hole and greedy path

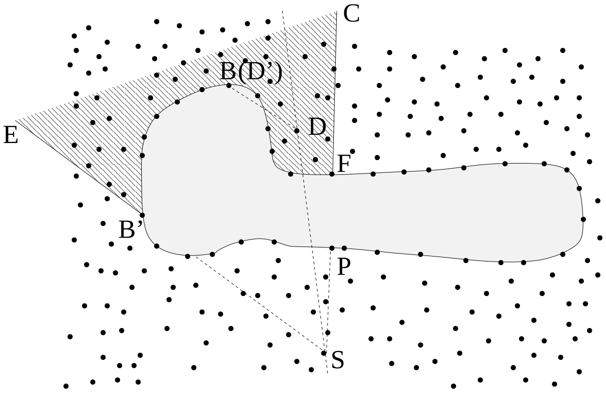

Figure 4.5: An example illustrating shaded area based on concave hole and non-greedy path

is hidden behind the hole. To avoid a detour path, S sends the packets to B first, and B will then relay them to D. Both paths can be greedy paths. If the path from S to B is not a greedy path, we may have to use recursion to find a landmark node for node B (Fig. 4.5). We observe that for some destination node $D' \in T$, the greedy forwarding from S to D' may be stuck at a different local minimum node (other than P). However, forwarding to B first can still benefit by having a shorter path than going through the local minimum node.

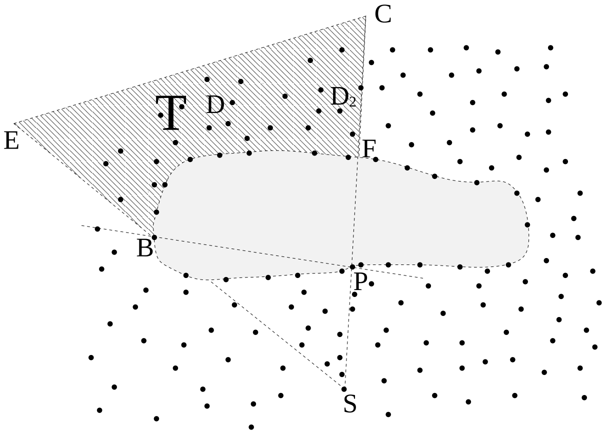

Figure 4.6: Illustration of general shaded area

Given a destination node D, we need to determine whether it is in the destination area T. As shown in Fig. 4.6, the area is enclosed by rays and partial edges of the hole polygon. To simplify the calculation, our first step is to extend the shaded area to include the area enclosed by line BP, line PF and arc BF because this area has no active nodes. The new destination area becomes the area semi-enclosed by $EBPC$,

After this extension, determining if a destination node D is in the shaded area becomes simple. If a destination node D satisfies the following conditions, it must be located in the shaded area.

1) D and P are located on the same side of line SB;

2) D and B are located on the same side of line SP; and

3) D and S are located on the opposite sides of line BP.

Suppose that the coordinates of nodes S, B and P are $S(x_s, y_s)$, $B(x_b, y_b)$ and $P(x_p, y_p)$, respectively. Line SB can be described by the following equation.

$$\frac{y - y_s}{y_b - y_s} = \frac{x - x_s}{x_b - x_s}.$$

It can be written as

$$(y_b - y_s)x - (x_b - x_s)y + (x_b y_s - x_s y_b) = 0.$$

Let $f_1(x, y) = (y_b - y_s)x - (x_b - x_s)y + (x_b y_s - x_s y_b)$. Suppose D's coordinates are $D(x_d, y_d)$. D and P are located on the same side of line SB if and only if $f_1(x_d, y_d) * f_1(x_p, y_p) > 0$. To include the case of D on line SB, we can use

$$f_1(x_d, y_d) * f_1(x_p, y_p) \geq 0. \tag{4.1}$$

Similarly, we can find the equations for line SP and BP. Suppose they are $f_2(x, y) = 0$ and $f_3(x, y) = 0$, respectively. Nodes D and B are located on the same side of line SP if

$$f_2(x_d, y_d) * f_2(x_b, y_b) \geq 0. \tag{4.2}$$

Nodes D and S are located on the opposite sides of line SP if

$$f_3(x_d, y_d) * f_3(x_s, y_s) \leq 0. \tag{4.3}$$

If all three conditions (4.1), (4.2) and (4.3) are met, node D is in the shaded area.

Further calculations to find a routing path may or may not be recursive, depending on the shape of the hole. For instance, in scenario one, when S intends to send a packet to D in the shaded area, S sends the packets to B first, and B will then relay them to D. The complete routing path is constructed by using greedy strategy.

However, if S intends to send a packet to D in the shaded area in scenario three, S considers the landmark B as its tentative target $D^{'}$, and then recursively finds another landmark $B^{'}$ as its tentative target. So the complete routing path is constructed by three greedy paths $SB^{'}$, $B^{'}D^{'}$ and $D^{'}D$.

4.2 ITGR Routing Scheme

In this section, we describe our routing approach, namely, ITGR. In ITGR routing, besides the source address S and the destination address D, a packet may contain a list of intermediate targets $< I_1, I_2, \cdots, I_k >$, which we refer to as an ITGR list for the rest of this section. We define the target $\mathcal{T}$ of a packet as either the first element on the ITGR list if the list exists, or the destination address D if the list does not exist. Similar to existing geographic routing schemes, a packet forwarded in ITGR routing can be either in greedy mode or perimeter mode. Theoretically it can use any existing perimeter routing algorithm when necessary. However, for simplicity of presentation, we assume that perimeter mode of GPSR is used. Therefore, perimeter mode will also be called GPSR mode in this chapter. As stated in GPSR routing, packets in GPSR mode will contain the location of the local minimum node P, at which point forwarding is changed from greedy mode to GPSR mode.

In ITGR routing, nodes will have a local cache with entries representing shaded areas. Each shaded area is in the form of $< P_i, B_i >$, where P_i is the location of a local minimum node and B_i is the location of the landmark node. In this paper, we define shaded area with entry $< P_i, B_i >$ saved in S the area swept clockwise by a ray starting from SB_i and ending at SP_i.

When source S needs to send a packet to destination D, it will call function *ITGR_send*() given in Fig. 4.7. *ITGR_send*() first gets the target $\mathcal{T}$ of the packet. It searches its local cache to see whether target $\mathcal{T}$ is in any of the shaded areas. If yes, it extracts the landmark node (B_1), corresponding to the target $\mathcal{T}$, and uses

```
ITGR_send()
  if the packet contains ITGR list
      $\mathcal{T}$ = first element of the ITGR list;
  else $\mathcal{T} = D$;
  Search local cache;
  if $\mathcal{T}$ is in a shaded area
      Extract the list of landmark nodes $< B_k, \cdots, B_1 >$;
      if ITGR list exists
          Prepend $< B_k, \cdots, B_1 >$ to the list;
      else Create ITGR list $< B_k, \cdots, B_1 >$;
      $\mathcal{T} = B_k$;
  Looks up whether there is a neighbor closer to $\mathcal{T}$;
  if true
      Forwarding packet using greedy mode;
  else
      Record the current node as the local minimum in the packet;
      Forwarding packet using GPSR mode;
```

Figure 4.7: ITGR Sending Algorithm

this landmark node B_1 as the destination and then determines if B_1 is in any shaded area. If it is, it gets the landmark node B_2 corresponding to target B_1. This process continues until a landmark node B_k that does not lie in any shaded area. Suppose the list of landmarks formed is $B_1, B_2, \cdots, B_k$. If the packet does not contain an ITGR list, it will create one with elements $B_k, B_{k-1}, \cdots, B_1$. If the packet has an ITGR list, these elements will be added in the front. We expect that, in most cases, this list contains only one element B_1. After that, we need to reset $\mathcal{T}$ to the value of the first element of the ITGR list.

As a last step, it forwards the packet to the neighbor that is closest to $\mathcal{T}$. If no neighbor is closer to $\mathcal{T}$ than itself, it will change the packet to GPSR mode and follow the GPSR rules for forwarding (including putting the address of the current node as the local minimum node in the packet).

Fig. 4.8 describes the ITGR forwarding algorithm. When a node receives a packet, it first checks whether its address is equal to destination D. If it is, the forwarding process ends. Otherwise, it will check whether there is an ITGR list and whether its address is equal to the first element on the list. If this is the case, it will remove itself from the list and then call *ITGR_send*() to send the packet to the next hop.

Next, depending on the forwarding mode of the packet, ITGR_forward() will process the packet differently. If the packet is in greedy mode, the algorithm will call ITGR_send() to forward the packet to the next hop. If the packet is in GPSR mode, the algorithm will do GPSR processing. Specifically, if the condition for changing to greedy mode is satisfied according to GPSR routing, [1] it will change the forwarding mode to Greedy. In addition to sending the packet, it will send a *landmark_exist_msg* to source S with the locations of the local minimum node P and its own (as the landmark). Otherwise, it will continue GPSR forwarding.

```
ITGR_forward()
    if its address is equal to destination $D$
        Forwarding is finished and exit;
    if ITGR list exists and its address is equal
            to the first element of ITGR list
        Remove its address from the list;
        Call ITGR_send() to send the packet to next hop;
    elseif the packet is in Greedy mode forwarding
        Call ITGR_send() to send the packet to next hop;
    elseif the packet is in GPSR mode forwarding
        Set the value of $\mathcal{T}$ as the target of the packet;
        if the current node has a neighbor closer to $\mathcal{T}$
            Send a landmark_exist_msg to source $S$ with
                local minimum node $P$ and its own address;
            Change to Greedy mode forwarding and call ITGR_send();
        else Continue GPSR forwarding;
```

Figure 4.8: ITGR Forwarding Algorithm

When the source receives *landmark_exist_msg*, it records it in its cache.

As a final note for the routing algorithm, we want to limit the number of entries in the local cache for each node. One approach we explore is to combine multiple entries into one if possible. This can happen when source node S sends to different destinations and each results in a cache entry being inserted. However, these shaded areas may overlap with each other. We have developed algorithms to determine the relative relations between shaded areas and to combine them accordingly.

---

[1] An example of such a condition is that the forwarding node finds out that one of its neighbors is closer to D than itself.

4.3 Combining Entries about Shaded Areas

If node S sends many packets to different destinations, several detour paths will be generated by basic routing strategy GPSR. In this way, multiple entries with the format $< LocalMinimum$, $Landmark >$ might be generated and saved in the cache of the node. We would like to merge them to save space and facilitate efficient entry lookup. So, once a node S sends packets to a destination and generates a new entry $< P$, $B >$, it looks up the entries in its local cache and update them. There are two situations S needs to handle. One is that S finds the existing entry in its cache with the same landmark B. The other is that S finds the existing entry in its cache whose landmark is not B. In the first situation, suppose S finds an entry $< P^{'}$, $B >$ existing in its cache. S then updates its entries as follows. [2]

Case1: $< P^{'}$, $B > \subset < P$, $B >$. This is the case in which B and P are on the opposite sides of $SP^{'}$ (Fig. 4.9). This scenario can be determined by the coordinates of these points as follows. Suppose the coordinates of points S, B, P and $P^{'}$ are $S(x_s, y_s)$, $B(x_b, y_b)$, $P(x_p, y_p)$ and $P^{'}(x_{P^{'}}, y_{P^{'}})$, respectively. Then equation of line $SP^{'}$ is:
$(y_{p^{'}} - y_s)x - (x_{p^{'}} - x_s)y + (x_{p^{'}}y_s - x_s y_{p^{'}}) = 0$
Let $g_1(x, y) = (y_{p^{'}} - y_s)x - (x_{p^{'}} - x_s)y + (x_{p^{'}}y_s - x_s y_{p^{'}})$. Nodes B and P are located on the opposite sides of line $SP^{'}$ if

$$g_1(x_b, y_b) * g_1(x_p, y_p) \leq 0. \tag{4.4}$$

S updates the entries by removing $< P^{'}$, $B >$ and inserting $< P$, $B >$.

Case2: $< P$, $B > \subset < P^{'}$, $B >$. This is the case in which B and $P^{'}$ are on

---

[2] For simplification, $< P$, $B >$ also represents the area determined by the entry $< P$, $B >$.

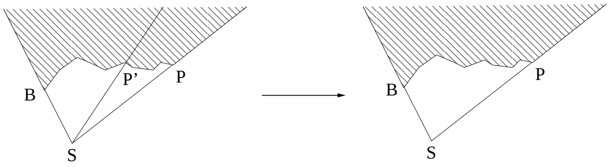

Figure 4.9: Entry update: Case 1.

the opposite sides of SP. (Fig. 4.10). We can also use coordinates of the points and equations of the lines to determine their relative locations. Because the existing entry $< P^{'}, B >$ covers the new entry $< P, B >$, S simply discards $< P, B >$.

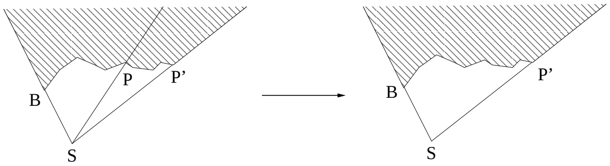

Figure 4.10: Entry update: Case 2.

The second situation is that S generates a new entry $< P, B >$. B and the existing landmark $B^{'}$ are different. We discuss the scenarios when $< P, B >$ overlaps with an existing entry $< P^{'}, B^{'} >$ in the local cache. Otherwise, S can simply insert the new entry. For instance, in Fig. 4.11, when S sends packets to D, its generates an area enclosed by EB, arc BF and line FC (the extension of SP), which is represented by SBP and associated with landmark B. When S sends packets to node $D^{'}$ later, it generates an area represented as $SB^{'}P^{'}$ and associated with landmark $B^{'}$. A naive update is to simply insert the new generated entry to S'*s* cache. However, the new entry and the existing entry have the common shaded area $EBF^{'}C^{'}$. Hence, if S wants to send packets to a node G located in the shared area, S will have two options of landmarks, $B^{'}$ and B (Fig. 4.11). Thus, S cannot determine its unique intermediate target. So S must keep all the areas stored in its cache to be disjoint. There are four cases to update $S's$ entries.

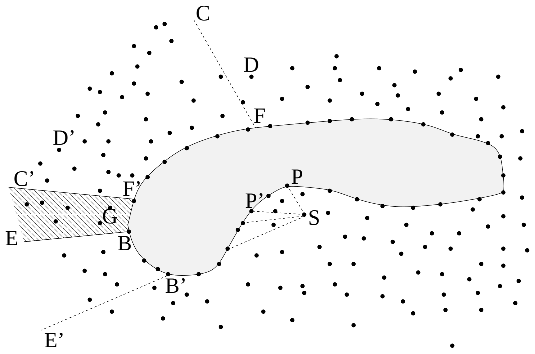

Figure 4.11: The case when two shaded areas are overlapped.

Case 1: $< P', B' > \subset < P, B >$. This is the case in which B and P are on the opposite sides of SB', and B and P are on the opposite sides of SP' (Fig. 4.12). The update is:

S removes $< P', B' >$ and then inserts $< P, B >$.

S does this update because $< P, B >$ fully covers $< P', B' >$.

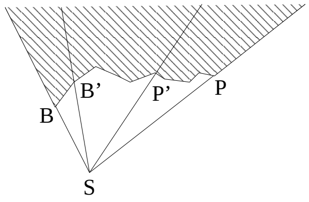

Figure 4.12: Combining entries: Case 1.

Case 2: $< P, B > \subset < P', B' >$ (Fig. 4.13). Under this scenario, S discards $< P, B >$ because the area determined by the new entry $< P, B >$ is covered by the existing entry $< P', B' >$.

Case 3: $< P, B >$ and $< P', B' >$ are overlapped. This is a case in which B and P are on the opposite sides of SB', and B and P are on the same side of SP' (Fig. 4.14).

The update is: S keeps the entry $< P', B' >$ and inserts a new entry $< B', B >$. S

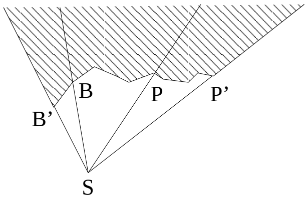

Figure 4.13: Combining entries: Case 2.

does this because the new entry $< P, B >$ can be considered as two area BSB' and $B'SP$. $B'SP$ is included in $B'SP'$, so only BSB' is inserted.

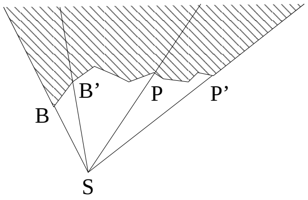

Figure 4.14: Combining entries: Case 3.

Case 4: $< P, B >$ and $< P', B' >$ are overlapped. This is a case in which B' and P' are on the opposite sides of SB, and B and P are on the same side of SB' (Fig. 4.15).

The update is: S removes the entry $< P', B' >$ and then inserts two new entries $< B, B' >$ and $< P, B >$.

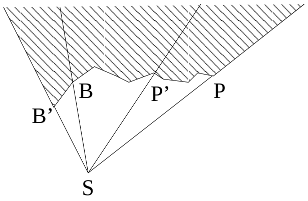

Figure 4.15: Combining entries: Case 4.

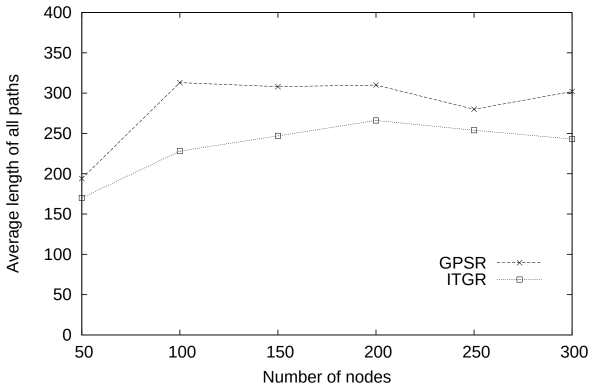

Figure 4.16: Performance evaluation: Comparison of ITGR and GPSR based on the average length of paths

4.4 Experimental Results

We conducted simulations using easim3D wireless network simulator [2]. We use a noiseless immobile radio network environment with an area of 400m×400m. Nodes have a transmission range of 40 meters.

We implemented both GPSR routing protocol and our ITGR routing protocol using this simulation model. Two metrics, namely, the length of routing path and the number of hops, are used for evaluating performance. The number of nodes (density) is varied from 50 to 300 in increments of 50. For each case, 10 connected networks were generated with void areas set inside the network.

Fig. 4.16 shows the average length of paths when the number of nodes changes from 50 to 300. The average length in ITGR is 17.52% shorter than that of GPSR when there are 50 nodes in the network. When the density network increases, ITGR performs even better. Fig. 4.17 shows the average number of hops with different node numbers. Similarly, the average number of hops in ITGR is 14.97% less than that of GPSR in the 50 node case. In Fig. 4.16, the average length of paths with 50 nodes (both GPSR and ITGR) is much smaller than other cases. This is because in the network plane, to guarantee the network's connectivity, 50 nodes have to be

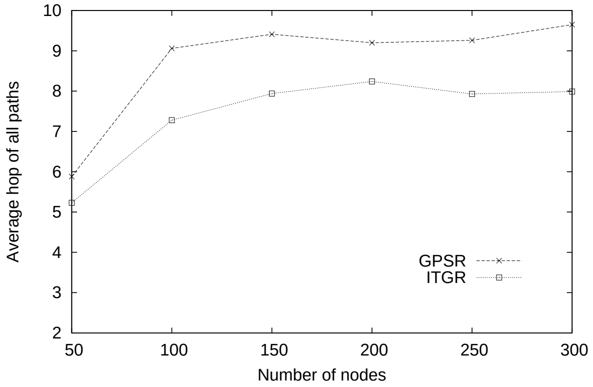

Figure 4.17: Performance evaluation: Comparison of ITGR and GPSR based on the number of hops

distributed in a relatively smaller area. This results in a shorter length and also a smaller number of hops.

To further illustrate ITGR's effect on the length of paths and the number of hops, we divide the tested paths into two types. For a routing path in ITGR routing, if no node in this path uses ITGR list for routing, we call this path a type 1 path. Otherwise the path is a type 2 path. We also collected the data for the paths when GPSR routing is used.

The percentage of type 2 paths over all paths is shown in Table 4.1. It ranges from 23.2% for a 50 node network and 16.2% for 300 node network. The larger the number of nodes in the network, the smaller the percentage. This is because the nodes are distributed in a plane with fixed sizes. The size of holes in sparse networks is larger than that in dense networks. Therefore, more paths are affected by void areas when the number of nodes is small.

Table 4.1: The average percentage of type 2 paths over all paths

Number of nodes	50	100	150	200	250	300
Percentage	23.2%	21.1%	18.7%	17.6%	16.4%	16.2%

Fig. 4.18 and Fig. 4.19 compare type 2 paths only. Compared with GPSR, ITGR

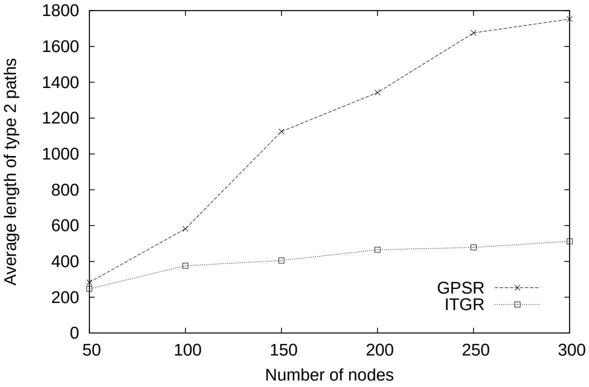

Figure 4.18: Performance evaluation: Comparison of ITGR and GPSR based on the average length of type 2 paths

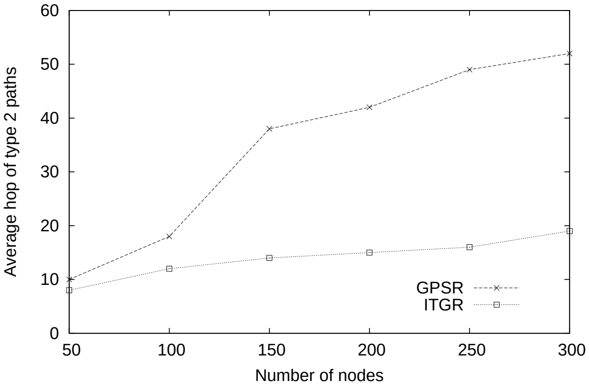

Figure 4.19: Performance evaluation: Comparison of ITGR and GPSR based on the number of hops of type 2 paths

generates much shorter paths and less hops. For type 2 paths, the average length of ITGR is only 29.5% that of GPSR and the number of hops is only 27.3% for 300 node networks. The gap between ITGR and GPSR increases when the number of nodes in networks increases. This is because when the number of nodes is larger, longer detour paths may exist for GPSR. For example, we find a path with 87 hops generated by GPSR when the number of nodes is 250. This path was reduced to 15 hops by ITGR. From these two figures, we can see that ITGR reduces length of paths significantly.

To evaluate the efficiency about how much the combining entries scheme compacts the entries, the average entries before and after combination based on different number of nodes are reported in Table 4.2.

Table 4.2: The report of combining entries

Number of nodes	50	100	150	200	250	300
Original number of entries	17.7	19.3	20.2	21.8	23.5	25.4
Number of entries after combining	7.2	9.6	11.1	13.2	14.8	16.1
Combining ratio	59.4%	50.3%	45.1%	39.5%	37.1%	36.7%

We define the number of reduced entries over the number of original entries as combining ratio. Table 4.2 illustrates that the combining ratio varies from 59.4% to 36.7% when the number of nodes increases from 50 to 300. The larger number of nodes corresponds to smaller ratio because when the number of nodes is larger, the entries are distributed to more nodes and then less combination operations are conducted. Table 4.2 also implies the recursive characteristics in ITGR. For instance, there are 16.2% paths that use the entries when the number of nodes is 300 in table 1. However, the original number of entries is 25.4. This is because some paths recursively use multiple entries.

Chapter 5

Hole Detection and Adaptive Geographic Routing in Ad Hoc Networks

In Chapter 4, we discussed an Intermediate Target Based Geographic Routing protocol to set up efficient paths when holes exist. In this chapter, we discuss another approach to handle the problem. This approach can detect holes ahead, thus preventing packets being forwarded towards a hole.

5.1 Outline of the Method

In Ad Hoc network deployment, we normally expect that the nodes can be distributed uniformly. However, there may still be regions with nodes density much lower than other regions. In addition, terrain variation and power depletion can also cause non-uniform node distribution. Therefore, Ad Hoc networks often have holes. In the routing process, a forwarding packet may get stuck because of the existence of holes. GPSR [26] can set up a path to bypass the holes in the network. Unfortunately, the path often requires a long detour.

We attempt to develop algorithms to overcome the local minimum issue in geographic routing by finding holes prior to packet forwarding towards the holes. Scholars may use a particular application to define and find a hole in some real work applications. For instance, in a sensor network that monitors temperature in a region, if we let a sensor node mark itself as unavailable once its local temperature exceeds a threshold, then the boundary of a hole can probably be determined based on the temperatures of the nodes. Such a hole is represented as a polygon that encloses all the sensors with local temperatures higher than the threshold. Unfortunately, these algorithms are time or space consuming. Moreover, the representation of a hole is too complicated. Most recent work tries to detect a hole and the nodes located on the hole's boundary in advance [31] [45]. The nodes on the boundary further advertise the hole information to some other nodes. In this way, the future routing path can be adaptive in the presence of the hole. In this chapter, we introduce an algorithm for Hole Detection and Adaptive Routing (HDAR) of Geographic Ad Hoc Networks. It focuses on defining and detecting holes in an Ad Hoc network, representing holes and building routes around the holes. It is a heuristic algorithm aimed to detect a hole quickly and easily. The hole can be identified by a constant time complexity calculation. In addition, we provide a very concise format to represent a hole by representing a hole as a segment. Moreover, we develop an approach to make part of the nodes located on the hole's boundary announce to the nodes in the vicinity of the hole. We further found the best trade-off between the overhead of hole information announcement and the benefit for future routing.

5.2 Hole Detecting Algorithm

5.2.1 Metric to Determine a Hole

In HDAR, a node p begins to detect whether it is located on the boundary of a hole only if the angle between its two adjacent edges is greater than 120 degrees [31].

p initiates a probe message which includes its location. p sends the message to its leftmost node with respect to the angle. The leftmost node can be defined as follows. p faces the area formed by the two rays of this angle, and uses the angle's bisector line to conduct counter-clockwise sweeping. The leftmost node is the first one that is met by the sweeping line. Upon receiving the probe message, p's leftmost neighbor node writes its location into the message and passes it to its leftmost neighbor. The probe message will finally come back to node p from p's rightmost neighbor with respect to the initial investigated angle [31] [45], where the right most neighbor is defined in the similar way as the leftmost node. When the probe message circulates, it collects the locations of the nodes on its way. So node p knows all the nodes' locations on the way.

p then begins to investigate the nodes by traveling clockwise from node to node. For each node on the way, p computes the length of their probe path $length\_pro()$ and their Euclidean distance $dist\_euc()$. For a node x, $length\_pro(p,x)/dist\_euc(p,x)$ is defined as hole detection ratio from p to x. If there exists a node v, the hole detection ratio from p to whom is larger than a predefined threshold δ, that is,

$$length\_pro(p,v)/dist\_euc(p,v) > \delta, \qquad (5.1)$$

then p is considered sitting on the boundary of a hole. The value of δ affects the hole detection results. If the value of δ is too large, it introduces false negatives. If δ is too small, it causes false positives. We derived that δ=2.25 is a good choice to detect most holes that will block greedy forwarding. Fig. 5.1 is an example for hole detection. Node p initiates the hole probe message. p collects the nodes' locations while the message circulates the loop. If p finds that there exists a node v, satisfying $length\_pro(p,v)/dist\_euc(p,v) > 2.25$, p is considered to be sitting on a hole.

The hole that is detected is a polygon. Note that some nodes located on the polygon measure they are located on the hole, but other nodes may not consider themselves on the hole. For instance, in Fig. 5.1, nodes g, p and h consider themselves

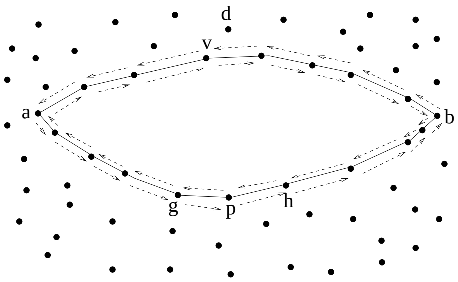

Figure 5.1: Illustration of the approach of hole detection

on the hole because there are nodes on the polygon's boundary that satisfy the hole definition for nodes g, p and h (5.1). However, nodes a and b at the hole polygon found by p do not consider themselves on a hole because there is no node on the hole satisfying condition (5.1) for nodes a and b. In fact, a or b's greedy forwarding will not be blocked by the polygon.

Any node located on the polygon may detect the hole repeatedly independently, thus a lot of overhead will be generated. We design a mechanism to reduce the redundant probes for discovering the hole. Once a node hears a probe message, it will not schedule a probe message although it has not sent out its probe message yet. In order to make each node know the location of every node on the polygon, the probe initiating node sends two probe messages at the same time, one clockwise and one counter-clockwise. (shown in Fig. 5.1). In this way, each node on the polygon can obtain all of the information of the polygon. Because the probe message is sent in both clockwise and counter-clockwise directions, there will be two probe paths. We choose the longer as the length of the probe path to calculate the hole detection ratio. We describe the probe message initiating algorithm in Fig. 5.2.

The probe message receiving algorithm is described in Fig. 5.3. In this algorithm, upon receiving a probe message, a node determines whether it is message initiation node. If it is, the node will calculate the hole detection ratio when both probe massages come back. If the node is not the message initiation node, it will write its

```
Probe_msg_initiating()
  if does not receive a probe message
    Search whether it has an angle between two adjacent edges
    larger than 120 degrees;
    if true
      Initiate a probe message;
      Write its location to this message;
      Send it to its left node and right node of the angle.
```

Figure 5.2: Probe Message Initiating Algorithm

location to the message and forward the message.

```
Probe_msg_receiving()
  Compare its location with the message initiator's location to
  determine whether it is the message initiator
  if it is the initiator
    Search whether the two probe message from
    different directions both reached it;
    if true
      Calculate the hole detection ratio
    else
      Wait for the second probe message;
  else
    Write its location;
    Forward the message to its left or right neighbor
    according to the forwarding direction.
```

Figure 5.3: Probe Message Receiving Algorithm

The probe initiator must have an angle between two adjacent edges with respect to it that is larger than 120° [31]. However, such an angle is necessary but not a sufficient condition to determine if the initiator is a local minimum node. In our algorithm, it does not matter whether the probe initiator is a local minimum node or not. The objective of the hole probe message is to find a hole, but not to determine if the probe initiator is a local minimum node.

Most likely, a probe initiator that finds a hole is a local minimum node. For example, in Fig. 5.1, node p initiates the probe message and finds that it is located on a hole. It is a local minimum node if it sends a packet to nodes in the vicinity of node d. However, it is not necessary for the probe initiator to be a local minimum node. For instance, in Fig. 5.4, node p initiates a hole probe message and detects a hole, but p is not a local minimum node because either its neighbor g or h is closer to any destination node in the area in the opposite side of ab. The hole information will be announced to the nodes in a certain area and these nodes will benefit from the hole announcement for future routing.

In Fig. 5.5, p is a local minimum node. p initiates a hole probe message but it cannot detect the hole because the length of the probe path from p to any node on the polygon over their Euclidean distance is approximate to 1. However, the hole can be detected by another node such as n and the hole information will be announced to nodes in areas (ekf and $e^{'}k^{'}f^{'}$) containing the nodes which will benefit from the hole information in future routing. This phenomenon indicates the scenario that a local minimum node p cannot detect a hole. This is because the polygon is long and narrow, and then the initiator's routing will not be blocked by the polygon. So this detecting result has very minor effect on p's forwarding. Nevertheless, the hole will be detected by another node who suffers from the hole.

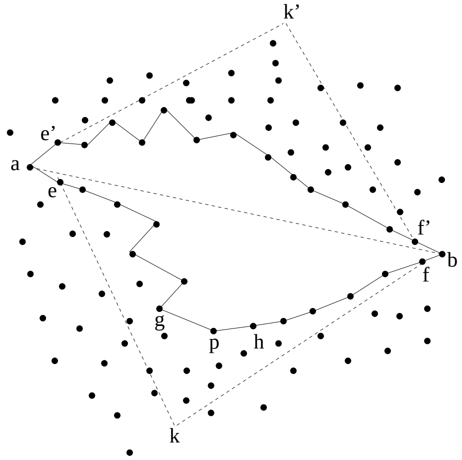

Figure 5.4: Hole detection scenario: Example 1

5.2.2 Shape-Free Hole Representation

A hole that is detected is a polygon. The representation of a polygon is a sequence of vertices. However, in geographic routing, we do not have to care about all the nodes

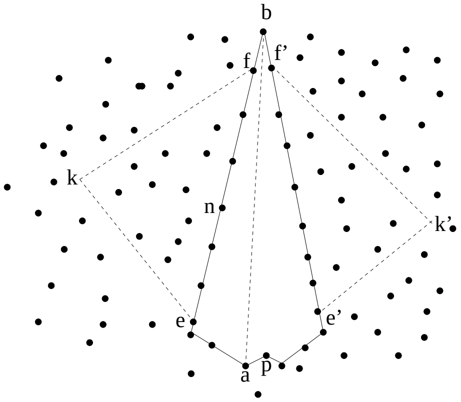

Figure 5.5: Hole detection scenario: Example 2

on the polygon because most of them have a minor effect on determining the routing paths. What we are interested are the nodes that will block the greedy forwarding. In our model, p can calculate the two nodes whose Euclidean distance is most remote because p has already obtained all the nodes' locations on the polygon (Fig. 5.4). The segment connecting these two nodes looks like a board that blocks greedy forwarding. For instance, segment ab in Fig. 5.4 is a board that blocks greedy forwarding. Then the hole is represented as $< a, b >$. No matter what the shape of the hole is, what we are concerned is the segment connecting the two most remote nodes. The size of the hole may change due to node failure or the addition of new nodes. In order to detect and represent the hole accurately, node p needs to send the information about the vertices lying on the polygon to nodes a and b for future detection of size changes of the hole.

The struck of greedy forwarding by the board $< a, b >$ is because some potential destination nodes are hidden behind the board, and source nodes located in a certain area on the opposite side of these hidden destinations are not aware of these destination nodes. In the basic routing approach, each node uses greedy forwarding until it fails due to a local minimum node, where greedy forwarding changes to perimeter forwarding. Thus the detour paths are generated. If the possible destination nodes hidden behind the board can be determined in advance and be announced to the source nodes unaware of these destination nodes, the lengths of the routing paths can be reduced dramatically. We determine the possible destination area (shaded area) as follows. Draw line ar perpendicular to segment ab, where r and p are on the opposite sides of ab. Also draw line bt perpendicular to line ab, where t and p are on the opposite sides of ab. Then the area $rabt$ is the shaded area (Fig. 5.6).

5.2.3 Hole Announcement

The nodes in area $rabt$ are the possible destination nodes for some source nodes. We would like to figure out an area containing these source nodes that need to be announced the hole information on the opposite side of $rabt$ (Fig. 5.6). The hole information can help the nodes adaptively adjust the next forwarding hops to avoid detour routing paths. In order to determine the hole announcement area, the announcement breadth and depth need to be figured out. We first determine two nodes e and f. They are the left and right nodes furthest away from each other at the same side as node p of segment ab, and satisfy the hole detection condition (5.1). Let c be the midpoint of segment ef. Draw segment ck perpendicular to ef. Then triangle efk is the area that should be announced the hole information. Note that if the hole announcement area is larger, more nodes will be benefited by the hole information and their future routing path will be shorter. At the same time, higher overhead will be introduced because more nodes need to be announced the hole information. So we would like to find a good balance between the benefit to future routing paths and the overhead. In our approach, the breadth of hole announcement is selected as segment ef because e and f are the most remote nodes on the hole's boundary located on the same side of p satisfying the hole detection condition. So the announcement depth determines the size of the area. We approximate $\triangle ekf$ by $\triangle akb$ (Fig. 5.7) because e and a are very close and so are f and b. We approximate local minimal node p

as the midpoint of ab. Let the length of ap be l and $\angle pak$ be α. In this way, the announcement depth is $\mid pk \mid$. It can be represented as $l\tan\alpha$ (5.2). It is directly proportional to the announcement overhead. For node k, if it sends a packet to node d on the opposite side of ab by GPSR, most likely, the entire path includes sub-paths $k \rightarrow p$, $p \rightarrow a$, and the path from a to d. On the contrary, if the entire path is set up by HDAR, it will include sub-paths $k \rightarrow a$, and the path from a to d. Hence, the benefit resulted by HDAR is the reduced length as follows:

$$
\begin{aligned}
&\mid kp \mid + \mid pa \mid - \mid ka \mid \\
&= l\tan\alpha + l - \frac{l}{\cos\alpha} \qquad (5.3)
\end{aligned}
$$

In order to find the balance of the size of announcement, we draw the curves of function 5.2 and 5.3 by dashed line and solid line respectively based on horizontal coordinate α in Fig. 5.8. The curve of overhead shows that before α is $60°$, overhead increases slowly. But, it increases fast after α is $60°$. The curve of benefit illustrates that after α is $60°$, benefit increases very slowly. In this way, we find that a good balance point between announcement overhead and benefit is where α is $60°$. At this point, $\mid pk \mid$ is $0.87* \mid ab \mid$ or $1.74l$. In this way, the depth of announcement is $1.74l$.

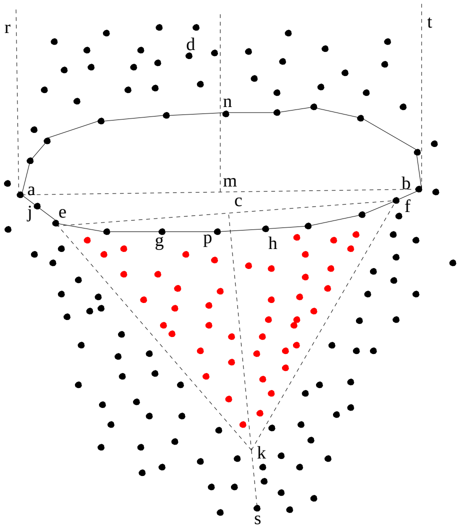

Figure 5.6: The area to be announced hole information

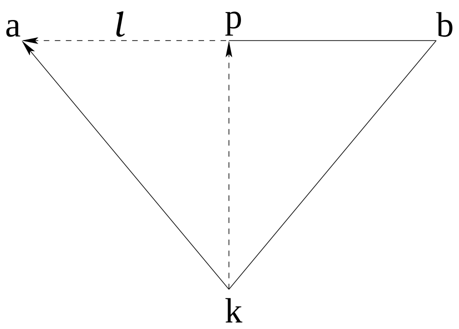

Figure 5.7: A triangle to represent hole announcement overhead and benefit

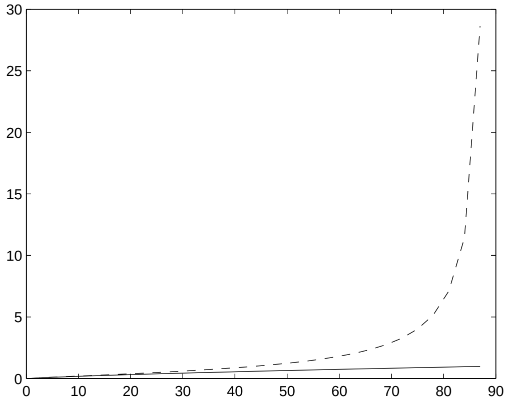

Figure 5.8: To find the balance point

The nodes on arc ef begin to advertise the hole information $< a, b >$ to their neighbors. In order to avoid duplicate messages, once a node in the area has received the hole information, it simply discards the duplicate. After the advertisement of the hole's information, each node in the area efk is aware of the hole $< a, b >$. Consequently, these nodes know that any possible destination node in area $rabt$ is hidden behind the hole. They should avoid packets being forwarded towards the hole in future routing (Fig. 5.6).

5.3 Adaptive routing

After the announcement, each node in the triangle ekf knows that there is a hole $< a, b >$ that blocks greedy forwarding to any destination node in area $rabt$. Thus the nodes in triangle ekf can adaptively adjust routing paths. In the network plane, once a node s intends to send a packet with destination d, it first looks up its local cache to see whether it has a hole information entry $< a, b >$. If there is no such entry in its cache, it just uses GPSR. Otherwise if there is a hole information entry $< a, b >$, but s and d are located at the same side of segment ab, s just uses GPSR. If s and d are located on the opposite sides of ab but d is not in the shaded area $rabt$, s just uses GPSR. Otherwise d lies in the area $rabt$. In this situation, s considers a or b as its tentative target. It writes a or b to the packet's header as a tentative target. In order to make s determine which one should be the tentative target, let m be the midpoint of segment ab, mn is perpendicular to segment ab and n is on the opposite side of ab relative to s. Then if d is located in area $ramn$, s writes a to the packet's head as its tentative target. If d is located in area $nmbt$, s writes b to the packet's head as its tentative target. When the packet reaches a or b, the tentative target will continue sending the packet to the destination node d.

When a node forwards a packet to its next hop, it calls procedure $HDAR\_forwarding()$ given in Fig. 5.9.

5.4 Experimental Results

We perform simulations using easim3D wireless network simulator [2], which is used to simulate IEEE 802.11 radios and is typically used for location based routing algorithms. We use a noiseless immobile radio network environment. In the simulations, nodes with a transmission range of 20 meters are deployed in an interest area of 400m×400m.

```
HDAR_forwarding()
  Look at the forwarding packet whether this node has a tentative
      target $\mathcal{T}$
  if true
      Compare whether this node is $\mathcal{T}$;
      if true
          Remove $\mathcal{T}$ and forward the packet to next hop with its
              destination $d$;
      else
          Forward the packet to next hop with its destination $\mathcal{T}$;
  else
      Search local cache
      if an entry $< a, b >$ exists
          if this node and $d$ are at the same side of $ab$
              Use GPSR;
          if this node and $d$ are at opposite sides of $ab$
              if $d$ is in $ramn$
                  Write $a$ as tentative target $\mathcal{T}$ to the packet;
                  Forward packet to next hop with destination $\mathcal{T}$;
              if $d$ is in $nmbt$
                  Write $b$ as tentative target $\mathcal{T}$ to the packet;
                  Forward packet to next hop with destination $\mathcal{T}$;
              else
                  Use GPSR;
      else
          Use GPSR.
```

Figure 5.9: Packet Forwarding Algorithm

We generate networks where the number of nodes varied from 50 to 300. For any given number of nodes, 50 networks are generated randomly. In each network, holes are generated automatically by the distribution of nodes.

Our experiments include two parts. The first part is to compare GPSR and HDAR. We implemented GPSR [39]. We compare GPSR and HDAR with respect to two metrics, the length of routing paths and the number of hops.

Fig. 5.10 shows the average length of paths when the number of nodes changes from 50 to 300. The average length in HDAR is 12.4% shorter than that of GPSR. Fig. 5.11 shows the average number of hops in HDAR is 13.2% less than that of GPSR.

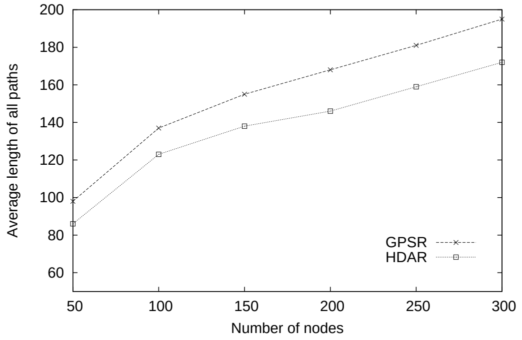

Figure 5.10: Performance evaluation: Comparison of HDAR and GPSR based on the average length of paths

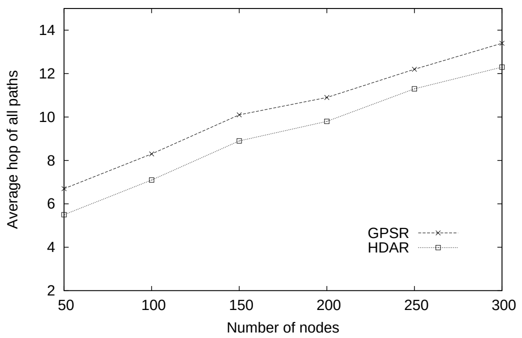

Figure 5.11: Performance evaluation: Comparison of HDAR and GPSR based on the average number of hops

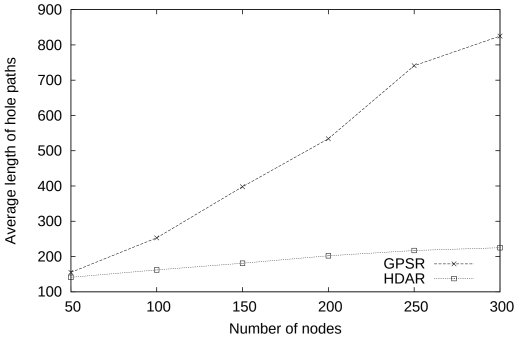

Figure 5.12: Performance evaluation: Comparison of HDAR and GPSR based on the average length of hole paths

In Fig. 5.10 and Fig. 5.11, the results report both the greedy path and the path in the vicinity of holes. In this way, HDAR's effect on the paths near the holes is not emphasized. To demonstrate HDAR's effect, we marked the paths that benefit from the hole information as "hole paths" and recorded the pairs of source and destination nodes. We also investigated the paths generated by GPSR with the same pairs of source and destination nodes. Then we compared the paths benefiting from hole information in HDAR with the paths derived from GPSR.

The performance of HDAR and GPSR for hole paths are reported in Fig. 5.12 and Fig. 5.13. HDAR has much shorter paths and fewer hops compared with GPSR. For the hole paths, the average length of HDAR is only 38.8% that of GPSR and the number of hops is only 35.4%. The two figures indicate that HDAR reduces the long detour paths around holes significantly.

The second part is to compare the computational complexity of HARG and our algorithm HDAR. We used the same networks as in part 1. Because HARG had not been simulated yet at the time of their publication [45], we selected $5\pi/6$ as the hole detection threshold and 60 meters as the diameter threshold. In both HARG and HDAR, we investigated the number of computation times of hole detection. In HDAR, the hole information is only calculated by a few nodes located on the

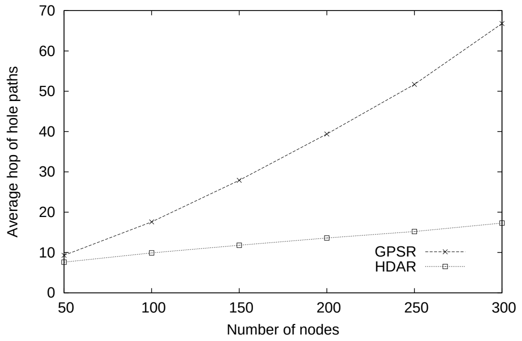

Figure 5.13: Performance evaluation: Comparison of HDAR and GPSR based on the average number of hops of hole paths

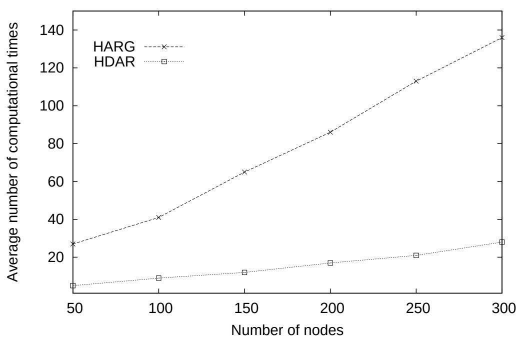

Figure 5.14: Comparison of HDAR and HAGR based on the computational complexity

hole and other nodes are advertised the hole information. In HARG, a number of nodes have to perform calculation to determine the existence of a hole. The numbers of calculations performed were reported to evaluate the computational complexity. Fig. 5.14 illustrates that the computational complexity of HDAR is much less than that of HARG.

5.5 Further Discussion of Holes

In this section, we discuss the method to figure out the value of δ, and the situations that will cause a false positive or a false negative when detecting holes. We will also discuss two types of special circumstances about hole representation.

5.5.1 Derivation of the threshold

In geographic routing, when a hole exists, there will be a detour path. So we attempt to detect a hole by finding a detour path. In our approach, "detour" path is defined as the routing path between two nodes that is much longer than their Euclidean distance. In order to quantitatively represent "much longer," we introduce a threshold δ for the ratio of routing length over Euclidean distance $length\_pro()/dist\_euc()$. To determine the value of δ, we first approximate the polygon by a circle (Fig. 5.15) in which δ is $\pi/2 = 1.57$. However, the circle is not a hole, so we will investigate $\delta > 1.57$.

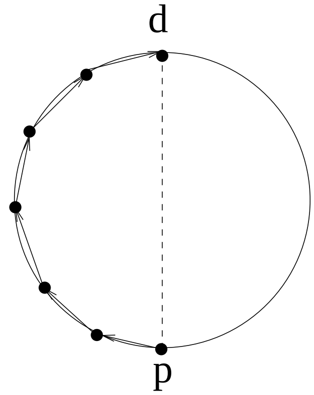

Figure 5.15: Illustration of the shape of a hole: scenario 1

We then increase the value of δ. Suppose that triangle abp is a equilateral triangle (Fig. 5.16), the length of each edge is 1, and the transmission range is slightly less than 1, such as 0.9. Then we move a to $a^{'}$ and let both $a^{'}p$ and $a^{'}b$ be equal to the transmission range. Then from p to b, a path $p \rightarrow a' \rightarrow b$ exists and it is a slight detour path. But the triangle is not a hole since none of the three nodes is a local minimum node. In this circumstance, the value of δ is approximate equal to 2.

Then we increase the value of δ to the one that is slightly larger than 2. We found that δ =2.25 is a good choice for a small false positive and a small false negative by experimental attempts.

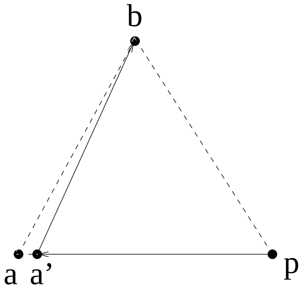

Figure 5.16: Illustration of the shape of a hole: scenario 2

5.5.2 False Negative and False Positive of Hole Detection

False negatives and false positives may occur during hole detection. Fig. 5.17 shows an example of false negative. In this figure, the transmission range is 0.9, $| pd |$=0.95, $| ad |$=$| bd |$=1,$| ed |$=$| df |$=0.9 and $| ap |$=$| pb |$=0.9. In this scenario, P cannot talk to d directly. Then p is a local minimum node and the polygon $paedfb$ is a hole. However, if p initiates a probe message, the distance of the probe path is 1.9. And the Euclidean distance is 0.95. The ratio of the two distances is 2, which is less than 2.25. Consequently, HDAR does not consider that polygon $paedfb$ is a hole. False negative is introduced by a very special circumstance that the detour path is between 2 to 2.25 long as the Euclidean distance, and the probe message initial node cannot talk to the destination directly. In network environment, the possibility of false negative is low. Moreover, false negative will not affect the routing too much because the detour is not too long, normally one hop longer than the Euclidean distance.

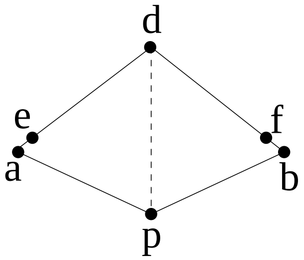

Figure 5.17: Illustration of a false negative when detecting holes by HDAR

False positive can also occur in some special circumstances. For instance, in Fig. 5.18, $\mid pd \mid=1$, $\mid ed \mid=0.95$, and $\mid ep \mid=\mid ea \mid=\mid ad \mid=\mid dc \mid=\mid cb \mid=0.9$. Once p wants to talk to d, p can find its neighbor e that is closer to d. Hence, p is not a local minimum node and then polygon $peadcb$ is not a hole. However, if p initiates a probe message, HDAR considers that the polygon is a hole. Furthermore, node e and p will advertise the hole information to an area epk. The area to be announced the hole information is not big because only a few nodes (two nodes in this example) announce the hole information. Although the polygon is a fake hole detected by HDAR, the nodes in epk will benefit from the hole information. For example, if s wants to send a packet to d, s will send the packet to d directly instead of a detour from p.

False negatives and false positives appear some time. However, their effects on HDAR algorithm is limited.

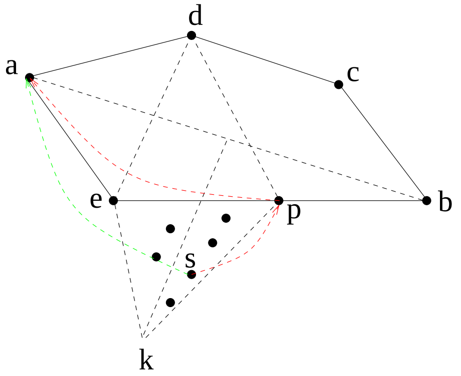

Figure 5.18: Illustration of a false positive when detecting holes by HDAR

5.5.3 Handling Size Changes of Holes

The size of a hole may change. Node failures may create additional holes in the network topology. Failure of a boundary node, especially the key nodes a or b in Fig. 5.19 can enlarge the size of the hole. To detect node failures, the boundary nodes that are a or b's neighbors send a beacon message to a or b periodically. For example, j sends a beacon message to a periodically. The time interval of the beacon

T is a system design parameter determined by application requirements. If node j has not heard the acknowledgment from a for several consecutive T intervals, it will inform node p to initiate a new hole probe message and figure out a new hole. Most likely, the new hole is larger than the previous one under this circumstance as shown Fig. 5.20.

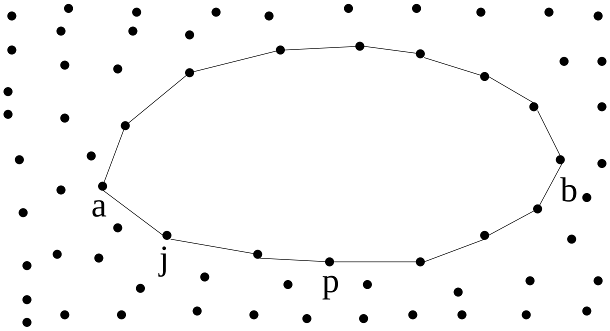

Figure 5.19: An example of hole ab

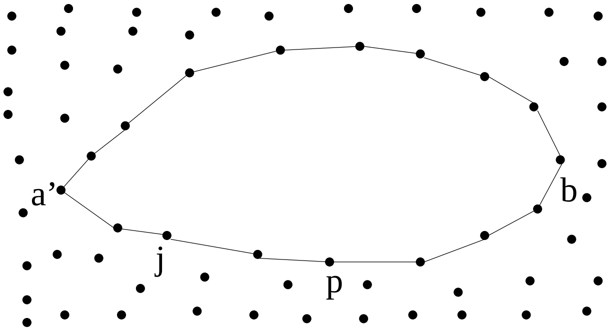

Figure 5.20: The enlarged hole $a'b$

It is possible that the size of the hole has decreased or even vanished due to newly added nodes which repair the hole. So nodes a or b need to send a beacon message to its neighbors periodically. If either one finds that a new node has been added, it will inform node p to initiate a new hole probe message and figure out the new hole. Most likely, the new hole is smaller than the previous one as shown in Fig. 5.21.

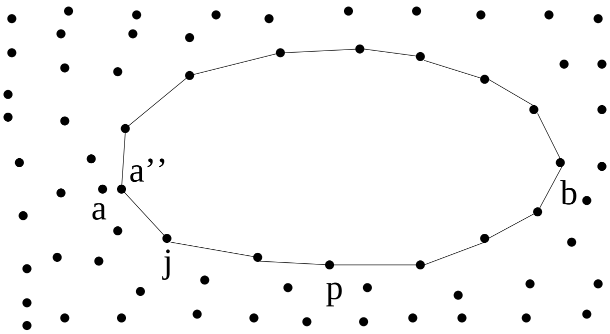

Figure 5.21: The decreased hole $a''b$

5.5.4 Analysis of Information Storage

Over the hole detection and announcement process, the majority of the nodes do not have to save any extra information. The nodes located on the boundary need to save the vertices of the hole polygon. The nodes located in the hole announcement area need to save the hole information, which is represented by two points. Hence, the cost of our algorithm is low in terms of space.

5.5.5 Handling Heterogeneous Transmission Ranges

In the previous chapter and this chapter, we assume that the transmission ranges of the wireless nodes are uniform. If the transmission ranges are non-uniform, we simply let two nodes claim each other as 1-hop neighbors only when both can hear from each other. Our algorithms can still be used.

Chapter 6

Statistical Filtering Based DTN Broadcasting for Vehicular Networks

In addition to transmitting packets over static nodes, we also need to discuss how to transmit packets on moving nodes. In this chapter, we discuss an approach that can broadcast messages over moving vehicles in inter-vehicle communication systems as fast as possible.

Inter-vehicle communication systems rely on multi-hop broadcasting to disseminate information to individual nodes beyond the transmission range. It is crucial to broadcast messages to other vehicles as fast as possible because the messages in vehicle communication systems are often emergency messages such as accident warnings or emergency vehicle sirens. The common approach in existing work is that the message initiator or sender selects a node among its neighbors that is farthest away from it in the broadcasting direction and then assigns the node to re-broadcast the message once the node gets out of its range or after a particular time slot. However, this approach may select a non-optimal candidate because it does not consider the moving status of vehicles including their moving directions and speeds. We develop a new approach based on Kalman Filtering, in which the current message sender can select the best candidate that will re-broadcast the message to other vehicles as fast as possible. Key to the decision making is to consider the candidates' previous moving status and predict the future moving trends of the candidates so that the message is spread out faster.

6.1 Kalman Filtering Based Broadcasting Protocol

6.1.1 Basic Idea

We start this section with a simple example to illustrate the basic idea behind our approach. We assume that all the vehicles (nodes) are distributed in a two dimensional space and they are moving. Each node has the same transmission range. In addition, errors of measurement for mobility exist because of environmental conditions [49].

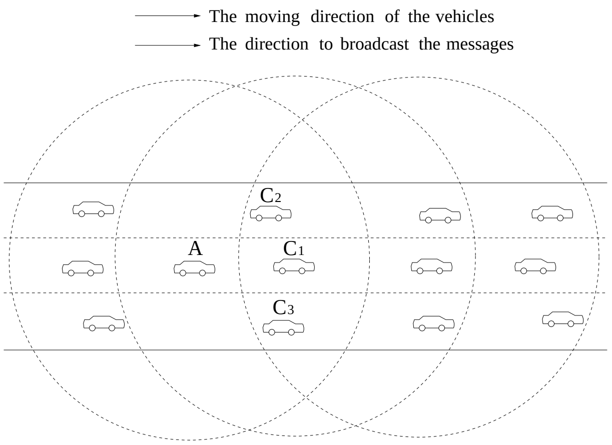

Figure 6.1: The basic broadcasting scenario

Fig. 6.1 is an example showing how a sender A should select one of its neighbors to re-broadcast its message. After node A broadcasts its message to all of its neighbors in the broadcasting direction, A's neighbors C_1, C_2 and C_3 receive this message. Node A needs to select one of them to re-broadcast the message. A simple strategy will let

A select the node farthest from A in the broadcasting direction. In this case, A will select C_1. However, this may not necessarily be the best choice.

Let us assume that node C_1 is moving at a much lower speed than C_2 and C_3. We also assume the historical velocities up to this moment of node C_2 are [..., 50, 55, 59, 63] (miles/hour), and the velocities of node C_3 are [..., 75, 71, 66, 64]. Our goal is to select a node that can re-broadcast the message fastest to other nodes. Considering that C_2 and C_3 are moving much faster than C_1, they are obviously better choices for carrying the message farther down the broadcast direction. If we only consider the current speed, A should choose C_3. However, we want to select the node that can get out of the transmission range of A and perform re-broadcasting first, thus achieving the goal of the fastest re-broadcast of the message. We can predict future velocities of these nodes based on their historical and current velocities, and then determine which node is the first one that will get out of A's range. In this example, our strategy will let A choose C_2.

6.1.2 Kalman Filtering Based Mechanism

The transmission range of the nodes in a vehicular network is normally 300-400 meters long. It is more than the width of the road containing multiple lanes. Thus we use a line to represent the road as in Fig. 6.2. Here partial arc l represents the transmission range crossing the road. In addition, point P is the intersection of l and the road.

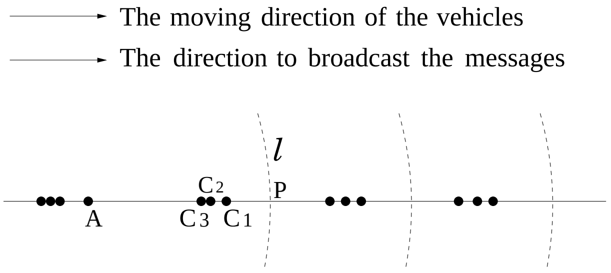

Figure 6.2: Simplified representation of the basic broadcasting

We need to compute which node is the first one that can get out of A's range, therefore moving trends must be considered. We come up with a model based on Kalman Filtering method [54]. In this model, each node only needs to maintain two parameters to represent its mobility trend; these parameters being its potential acceleration and its variance of acceleration. With this data, a node's future velocity can be predicated. Thus, node A can determine the optimal node assignment to perform the re-broadcasting subsequently.

We observe position at every time interval Δt. Suppose we are investigating the nodes at current time k. Node A would like to find the node that will reach P first and assign it as the re-broadcasting node. We mark the transmission range of A as R, the velocity of node C_i measured at time k as v_{i_k}, and the distance from A to C_i as $|AC_i|$. Then the time to reach P is $T_i = (R - |AC_i|)/|\hat{v_{i_k}}|$, where $\hat{v_{i_k}}$ is the predicted velocity of C_i at time k. The prediction is based on considering both C_i's historical and current mobility status. In our approach, we predict C_i's velocity for time $k+1$ to compute T_i. Once we know each $\hat{v_{i_k}}$, we will determine the node with the smallest T_i. So the task is to compute $\hat{v_{i_k}}$.

A naive method of predicting velocity at time $k+1$ is as follows. Let node C_i save all its historical and current (time k) velocities v_{i_0}, v_{i_1}, ...,v_{i_k}. Then it can predict its velocity for time $k+1$ by the curve-fitting method that approximates its moving trend. A problem with this method is its associated overhead. So we devise a model based on the stochastic filtering approach, particularly the Kalman Filtering method. In our model, each node only needs to maintain the values of two parameters, potential acceleration and variance of acceleration at time $k-1$, then measure the status of these parameters at time k.

We define the following notations to represent the mobility status for node C_i in Table 6.1. Note that all the notations of acceleration and velocity are vectors.

At time k, C_i measures its velocity v_{i_k}. And then it computes a_{i_k} as its measured

Table 6.1: Notations for the Mobility Status

Notation	Meaning
v_{i_k}	The measured velocity at time k.
$\hat{v_{i_k}}$	The velocity predicted at time k.
a_{i_k}	The acceleration measured at time k.
$\hat{a_{i_k}^{-}}$	The acceleration at time k evolved from time $k-1$.
$\hat{a_{i_k}}$	The potential acceleration at time k.
p_{i_k}	The variance of acceleration updated at time k.
$p_{i_k}^{-}$	The variance of acceleration at time k evolved from time $k-1$.
Q	The error or noise in the process.
Kg_{i_k}	The blending factor at time k.

acceleration by

$$a_{i_k} = (v_{i_k} - v_{i_{k-1}})/\Delta t$$

Node C_i updates $\hat{a_{i_k}^{-}}$ and $p_{i_k}^{-}$ in order to keep its historical moving trend to predict its future velocity.

$$\hat{a_{i_k}^{-}} = a_{\hat{i_{k-1}}}$$

$$p_{i_k}^{-} = p_{i_k-1} + Q$$

Node C_i also computes the blending factor Kg_{i_k}, which indicates the change in acceleration from the last time to the current time.

$$Kg_{i_k} = p_{i_k}^{-}(p_{i_k}^{-} + Q)^{-1} = p_{i_k}^{-}/(p_{i_k}^{-} + Q)$$

Once C_i obtains the blending factor Kg_{i_k} and the evolved acceleration $\hat{a_{i_k}^{-}}$, it knows the change in acceleration and the evolved acceleration. Additionally, C_i considers the measured acceleration a_{i_k}. Then it computes its potential acceleration $\hat{a_{i_k}}$. This acceleration will be used to predict its velocity for time $k+1$.

$$\hat{a_{i_k}} = \hat{a_{i_k}^{-}} + Kg_{i_k}(a_{i_k} - \hat{a_{i_k}^{-}})$$

C_i updates the variance of acceleration for future utilization.

$$p_{i_k} = (1 - Kg_{i_k})p^-_{i_k}$$

Finally, the velocity predicted for time $k+1$ is

$$v_{i_{k+1}} = v_{i_k} + \hat{a_{i_k}}\Delta t$$

In the entire computation process at time k, C_i only needs to measure its velocity v_{i_k} and record two parameters $\hat{a^-_{i_{k-1}}}$ and $p_{i_{k-1}}$. Then it predicts its velocity at time $k+1$ by the calculation.

The direction of the road may not be parallel or perpendicular to the coordinate axes. Moreover, a vehicle may suddenly change its moving direction or speed. So we decompose the velocity into two velocities parallel to the axes of the coordinate plane (Fig. 6.3) to take care of the general scenarios.

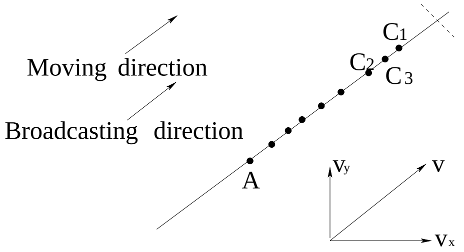

Figure 6.3: An example of coordinates decomposition

In the general scenario, node C_i computes its velocity $\hat{v_{i_{k+1_x}}}$ in the X direction and $\hat{v_{i_{k+1_y}}}$ in the Y direction at time $k+1$, then computes $\hat{v_{i_{k+1}}}=\sqrt{\hat{v_{i_{k+1_x}}}^2 + \hat{v_{i_{k+1_y}}}^2}$. It further computes T_i by $T_i = (R - |AC_i|)/|\hat{v_{i_{k+1}}}|$, the time that node C_i gets out of the range of A. Then A can figure out the desired node C_i with minimal T_i.

From node A's perspective, the moving direction of the vehicles and the message broadcasting direction can be identical or opposite. In addition, node A can move in a one-way road or a two-way road. In order to determine the type of a road, node A periodically sends beacon messages to its neighbors and collects the velocity information from its neighbors. If the velocities of all vehicles are in the same direction during a certain time duration, then node A knows it is moving on a one-way road. Once the node receives neighbor information with velocities in the opposite direction, node A knows it is on a two-way road. Suppose A is broadcasting a message. Its essential task is to find the best re-broadcasting node. Node A calls SFBB() to handle this task. Two cases, one-way road and two-way road are considered in SFBB() respectively.

Case 1: A is moving on a one-way road.

Scenario 1: When its moving direction and the broadcasting direction are identical (Fig. 6.4), node A selects the node that can get out of A's range in the moving direction firstly as the re-broadcasting node. For instance, A may designate node B to re-broadcast the message. Once B is designated, it calls Carry_and_broadcast(). If A cannot find such a node, that is, A is much faster than all other nodes, A carries the message and records the most remote point of its transmission range in the broadcasting direction. Once it reaches that point, it resumes broadcasting.

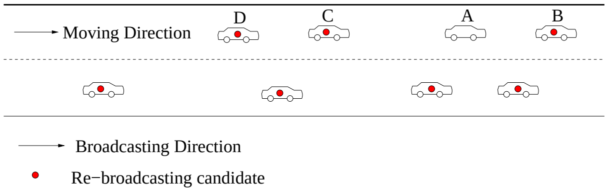

Figure 6.4: The scenario that moving direction and broadcasting direction are identical on a one-way road

Scenario 2: When its moving direction and the broadcasting direction are opposite (Fig. 6.5), node A selects the node that can get out of A's range in the moving direction firstly as the re-broadcasting node. For instance, A may select node E as the best candidate to re-broadcast the message. Once E is designated as the re-broadcaster, it calls Carry_and_broadcast(). If A cannot find such a node, that is, A is much slower than all other nodes, A simply calls Dist_selection() to determine the re-broadcasting node.

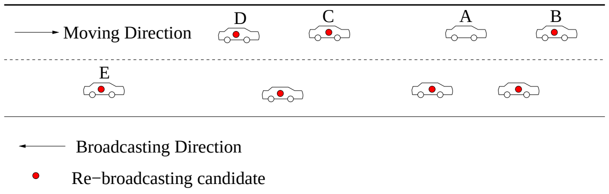

Figure 6.5: The scenario that moving direction and broadcasting direction are opposite on a one-way road

Case 2: A is moving on a two-way road.

Scenario 1: When its moving direction and the broadcasting direction are identical (Fig. 6.6), node A selects the node that can get out of A's range in the moving direction firstly as the re-broadcasting node. For instance, A may designate node B to re-broadcast the message. Once B is designated, it calls Carry_and_broadcast(). If A cannot find such a node, that is, A is much faster than all other nodes, A carries the message and records the most remote point of its transmission range in the broadcasting direction. Once it reaches that point, it resumes broadcasting.

Scenario 2: When its moving direction and the broadcasting direction are opposite (Fig. 6.7), node A investigates two candidates. One is the node in the opposite moving direction that can get out of A's range in the broadcasting direction first, such as node E in Fig. 6.7. The other is the node in its same moving direction that can get out of A's range in the broadcasting direction first, such as node D in Fig. 6.7. Then A compares the two nodes and finds the one that can get out of its range faster. Such a node will be assigned to preform re-broadcasting task. If A cannot find any node that gets out of A's range in the broadcasting direction, A simply calls Dist_selection() to determine the re-broadcasting node.

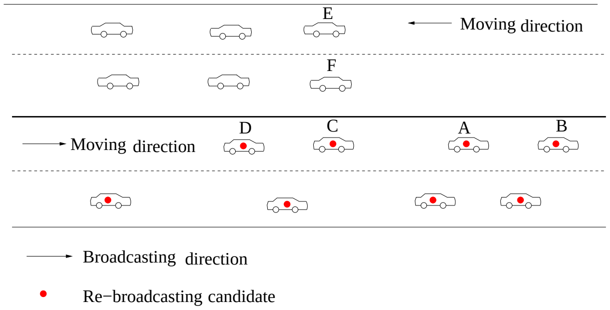

Figure 6.6: The scenario that moving direction and broadcasting direction are identical on a two-way road

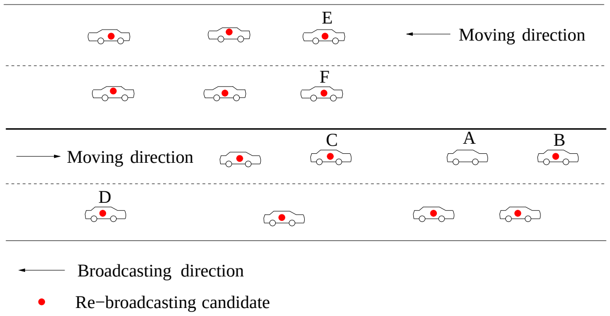

Figure 6.7: The scenario that moving direction and broadcasting direction are opposite on a two-way road

6.1.3 Algorithm Description

Dist_selection() is an existing algorithm. A node calls this procedure and finds the node that is farthest away from it in the broadcasting direction, and then designates the node to re-broadcast the message. It does not consider any moving status.

```
Dist_selection()
 if it finds a node $B$ that is farthest away from it in the broadcasting direction
    Send a re-broadcasting signal to $B$;
 else Carry the message until it finds such a node $B$.
```

Figure 6.8: Dist_selection algorithm

SFBB() can be described as follows.

```
SFBB()
 if it is moving on a one-way road
   if its moving direction and broadcasting direction are identical
     if it finds a node $B$ that gets out of its range earliest in the broadcasting direction
        Ask $B$ to perform carry_and_broadcast()
     if it cannot find such a node
        It records the border point of its range in the broadcasting direction
        Resume re-broadcasting when it reaches the border point
   else % its moving direction and broadcasting direction are opposite
     if it finds a node $B$ that gets out of its range earliest in the broadcasting direction
        Ask node $B$ to perform carry_and_broadcast()
     if it cannot find such a node
        Perform Dist_selection()
 else % it is moving on a two-way road
   if its moving direction and broadcasting direction are identical
     if it finds a node $B$ that gets out of its range earliest in the broadcasting direction
        Ask $B$ to perform carry_and_broadcast()
     if it cannot find such a node
        It records the border point of its range in the broadcasting direction
        Resume re-broadcasting when it reaches the border point
   else % its moving direction and broadcasting direction are opposite
     It considers its neighbors in both moving directions
     if it finds a node $B$ that gets out of its range earliest in the broadcasting direction
        Ask node $B$ to perform carry_and_broadcast()
     if it cannot find such a node
        Perform Dist_selection()
```

Figure 6.9: SFBB algorithm

In the broadcasting process, the location of the initiator is included in the broadcasting message. Once a node is designated as the re-broadcasting node, it computes its distance to the initiator. We define a distance threshold and only messages within this threshold are broadcast. If the distance is greater than the threshold, the node stops re-broadcasting. Otherwise it carries the message until it gets out of the message sender's transmission range and it has neighbors. Then it re-broadcasts the message to its neighbors and selects the one that will perform the re-broadcasting task.

Upon receiving a re-broadcasting message from another node, the node will call Carry_and_broadcast() if the distance to the message initiator is less than the threshold.

```
Carry_and_broadcast()
  if the node gets out of the range of the message sender
     and has neighbor(s) in the broadcasting direction
      Broadcast the message.
```

Figure 6.10: Carry_and_broadcast algorithm

6.2 Experimental Results

We conduct simulations using the GloMoSim wireless network simulator [60]. We use the tool BonnMotion [61] to generate mobile nodes. The mobility mode we use is "ManhattanGrid," which can generate nodes distributed in areas similar to roads or streets. The size of the environment is $2000\times2000\ m^2$. The number of vehicles varies from 50 to 300 with an increment of 50. We set the average packet generation rate from 0.1 to 0.9 per second. The packet generation rate represents the number of packets generated by a node per second. The MAC protocol is IEEE 802.11.

Many broadcasting protocols are based on the idea that the current sender selects the node that is farthest away from itself to conduct the re-broadcast task. This method is called "DIST" in the comparison. We compare our statistical filter based broadcast protocol "SFBB" with the "DIST" method using two metrics. The first one is the package delivery ratio, which is the ratio of the number of vehicles that receive the broadcasting message over the total number of vehicles. The second metric is the end-to-end delay. It is the time elapsed from packet generation to packet reception by the receiver in the desired area.

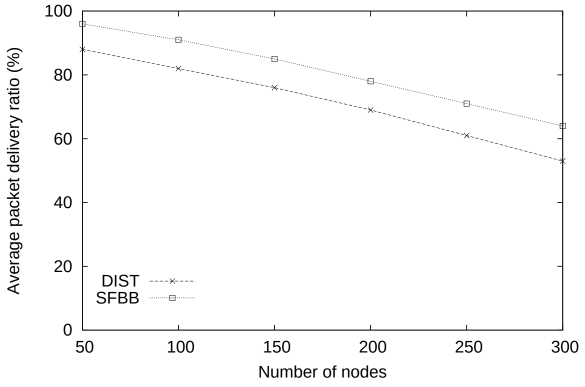

Figure 6.11: Comparison of packet delivery ratio between DIST and SFBB based on variation of the number of nodes

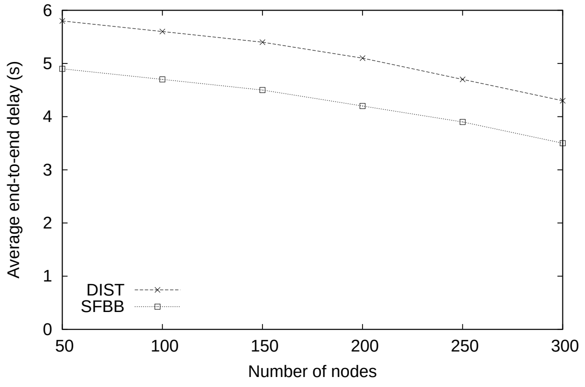

Figure 6.12: Comparison of end-to-end delay between DIST and SFBB based on variation of the number of nodes

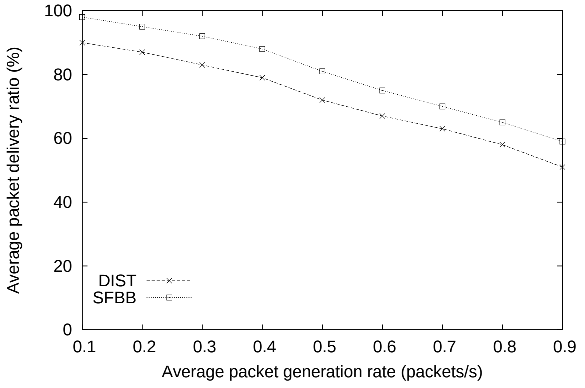

Figure 6.13: Comparison of packet delivery ratio between DIST and SFBB based on variation of the packet generation rate

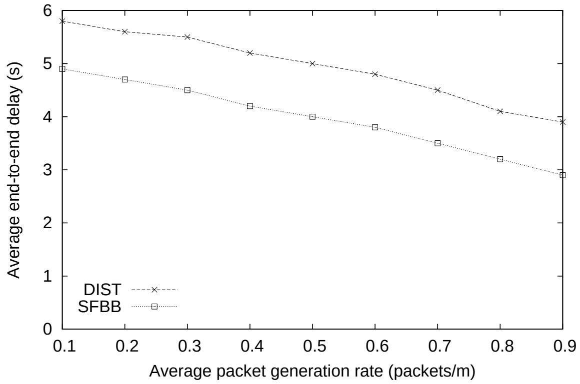

Figure 6.14: Comparison of end-to-end delay between DIST and SFBB based on variation of the packet generation rate

Fig. 6.11 shows the average packet delivery ratio when the number of nodes changes from 50 to 300 and the average packet generation rate is fixed at 0.5. The delivery ratio decreases when the number of nodes increases. This is because 802.11 protocol performs poorly with more nodes due to packet collision. We can see that the ratio generated by SFBB is 9.2% higher than DIST overall because SFBB can broadcast a message in a new place faster with less collision than DIST.

Fig. 6.12 shows the average end-to-end delay with the same configuration as Fig. 6.11. It illustrates the end-to-end delay of SFBB is 9.1% lower than that of DIST overall. This is because SFBB can select the fastest node to re-broadcast the message.

Fig. 6.13 presents the packet delivery ratio when the packet generation rate varies from 0.1 to 0.9 and the number of nodes is fixed at 150. Similar to Fig. 6.11, the ratio decreases when the value of the packet generation rate increases. This figure indicates that the delivery ratio of SFBB is 9.4% higher than DIST overall because the re-broadcasting nodes selected by SFBB enter a new place with less collision faster than DIST. Fig. 6.14 shows that the end-to-end delay of the two approaches. Again, SFBB has a smaller end-to-end delay than the DIST method. Overall SFBB is 9.0% lower than DIST.

Chapter 7

Conclusion and Future Work

This dissertation presents my research work on building efficient algorithms for channel allocation in wireless mesh networks, and designing efficient routing protocols in wireless and mobile network systems. In this chapter, I summarize my dissertation work and present my future research plan.

7.1 Research Accomplishments

Contributions of this dissertation can be summarized in four aspects. First, we developed a bipartite graph based algorithm for dynamic channel allocation in wireless mesh networks. The algorithm can yield near maximal bandwidth utilization and dramatically lower the possibility of starvation on channel assignment behavior. Second, we designed an efficient geographic routing protocol, ITGR. The novelty of ITGR is that it reduces the lengths of routing paths only by utilizing a single forwarding path experience. Third, we developed an efficient and effective geographic routing protocol, HDAR. We designed an effective algorithm with low complexity to detect holes and proposed an adaptive routing algorithm to set up the shortest path. Last, we developed a novel message broadcasting approach in vehicular networks based on Kalman Filtering, in which the current message sender can select the best candidate that will re-broadcast the message to other vehicles as fast as possible. The contribution is that during decision making, our algorithm considers the candidates' previous moving status and predicts the future moving trends of the candidates so that the message is propagated faster.

Dynamic Channel Allocation in Wireless Mesh Network

A wireless mesh network (WMN) is a communication network made up of network nodes organized in a mesh topology. The capacity of a wireless mesh network can be improved by equipping mesh nodes with multi-radios tuned to non-overlapping channels. We developed a new bipartite-graph based model and designed a channel allocation algorithm taking care of both aspects [59]. Our solution devised an augmenting path to find the local best matching iteratively in the bipartite graph until the global best matching is achieved. It yielded near maximal bandwidth utilization and reduced the possibilities of starvation on channel assignment behavior. Moreover, we used simulations to evaluate our algorithm and our experimental results demonstrated that it outperforms previous algorithms.

Intermediate Target Based Geographic Routing

Geographic routing is an emerging technology in Ad Hoc networks. However, when there are void areas (holes), the well-known GPSR protocol [26] often causes long detour paths. We proposed an innovative routing protocol-Intermediate Target Based Geographic Routing (ITGR) to solve this problem [37]. Under this protocol, the source determines areas containing many possible destination nodes which are shaded by the holes based on previous forwarding experience. The source then selects an intermediate node as the tentative target and greedily forwards packets to it, which in turn forwards the packet to the final destination by greedy routing. Our approach has the novel ingredient that a single forwarding path can be used to determine a shaded area that may cover many destination nodes. Simulation results demonstrated that this scheme significantly suppresses the long detour paths with much less overhead, compared with existing geographic routing protocols.

Hole Detection and Adaptive Geographic Routing for Ad Hoc Networks

In geographic routing, holes can be detected prior to routing to prevent packets from approaching the hole, thus reducing the length of a routing path. However, the consequent problem of this method is that many schemes turn out to have very high computational complexity. We proposed a novel protocol called Hole Detection and Adaptive Geographic Routing [38]. It identifies the hole efficiently only by comparing the length of routing path with the length of Euclidean distance between a certain pair of nodes. We then proposed a concise representation of the hole and developed an effective routing scheme to set up the shortest path based on it. Experiments have shown the expected low computational complexity of our approach.

Statistical Filtering Based DTN Broadcasting Protocol for Vehicular Networks

In vehicular networks, it is essential to broadcast messages to other vehicles as fast as possible. We developed a new approach based on Kalman Filtering [67]. In this protocol, the current message sender can select the best candidate to re-broadcast the message to other vehicles as fast as possible. The decision is based on the candidates' previous moving status and prediction of the future moving trends of the candidates so that the message is propagated faster. Simulations demonstrate that our approach can significantly decrease the end-to-end delay and improve the message delivery ratio.

7.2 Future Work

In the future, I want to expand my dissertation work in several directions.

I would like to extend my channel allocation and geographic routing algorithms to multi-metrics based routing in wireless mesh networks. Existing routing protocols of wireless networks typically find routes with the minimum count of hops or length of path. However, many of these protocols have poor throughput. As a result, minimum hop-count routing often chooses routes that have less capacity than the best paths that exist in the network. Observations suggest that more attention be paid to link bandwidth when choosing routing routes. Inspired by the success of my dynamic channel allocation mechanism and geographic routing protocols, I intend to develop routing protocols that choose high-quality routes. One possible approach is to assign weight representing corresponding bandwidth to each link, and then investigate both the length of path and the weight of link to find the best routing path.

I would be excited to further exploit a broad range of topics in vehicular networks. There are three underlying motivations for my future research. Firstly, I will pursue Spatiotemporally Mobicast Routing Protocol (SMRP) in the near future. Unlike ordinary mobicast routing protocols, the SMRP takes the time factor into account. Secondly, I will extend broadcasting from emergency information to comfort information applications, such as weather forecast, advertisements, and navigation information. Most current work focuses on transmitting emergency messages. I would like to pay extra attention to the transmission of comfort information in addition to emergency messages. Normally, channels are used for comfort message transmission. However, when emergency messages need to be transmitted, channels will be allocated to those messages. This topic is also closely linked with my dynamic channel allocation research topic. Thirdly, I plan to perform research in the low network density scenario. The existing broadcasting routing protocols are reliable for dense traffic scenarios. However, the broadcast message in the future should be able to disseminate under low network density as well, such as the sparse network in off-peak hours. In order to get accurate models of the above topics, my research would utilize a diversity of techniques including analysis of driving data, theoretical analysis, and vehicle behavioral experiments.

I will also conduct research on the security of ubiquitous mobile computing. Mobile computing is now everywhere and is human-oriented, embracing human-life as its computing subjects. Such ubiquitous computing can improve people's quality of life in terms of their health, social activities, and the environment. However, protecting the privacy and security of the user is of paramount importance. For instance, mobile nodes interact with the world continuously, this may reveal certain information about the users. Moreover, wireless transmissions expose the signal of legitimate users over a large network region, thereby opening up the possibility of several types of security attacks. We will focus on designing a reliable framework and protecting data integrity in mobile systems by using cryptographic protocols while minimizing its impact on the efficiency of mobile nodes. In addition, I will develop an automatic verification module in the secure protocols, and provide quantitative guarantees of security for network protocols, for example, how many times a particular key should be used.

I am also interested in further conducting DTN routing in vehicular environments. My research will focus on DTN routing for moving destinations. In the vehicular network scenario, the nodes are often moving. Traditional DTN routing only takes care of the partition phenomenon of networks. We will consider both the disconnections and the location change of the destination node. Two possible approaches can be used to handle the moving destination. The first one is passive tracking. Packets are forwarded towards the destination location. If the destination has moved away, the intermediate nodes will forward the packets along the moving route of the destination. The second one is active prediction. The source node and intermediate nodes predict the trajectory of the moving destination before packets are forwarded to it. In this way, packets are forwarded to the potential location of the destination. Statistical methods can be used to solve the prediction problem.

In summary, my future research will focus on designing algorithms which can achieve efficient routing by considering multiple metrics, security and Delay-Tolerant in wireless networks.

Bibliography

[1] I.F. Akyildiz and X. Wang, A Survey on Wireless Mesh Networks, *IEEE Communications Magazine*, 2005.

[2] C. Liu, Simulator in Department of Computer Science and Engineering at Florida Atlantic University.

[3] I. F. Akyildiz, X. Wang b, and W. Wang, Wireless Mesh Networks: A Survey, *Computer Networks Journal (Elsevier)*, Vol. 47, P445-487, 2005.

[4] Q. Zhao and B. Sadler, Dynamic Spectrum Access: Signal Process, Networking, and Regulatory Policy, *IEEE Signal Processing Magazine*, September, 2005.

[5] B. Ko, Distributed Channel Assignment in Multi-Radio 802.11 Mesh Networks, *Columbia University Technical Report*.

[6] R. Ashish, G. Kartik and C. Tzicker, Centralized Channel Assignment and Routing Algorithms for Multi-Channel Wireless Mesh Networks, *Mobile Computing and Communications Review*, 2005.

[7] K. N. Ramachandran, E. M. Belding, K. C. Almeroth, and M. M. Buddhikot, Interference-Aware Channel Assignment in Multi-Radio Wireless Mesh Networks, *IEEE Infocom*, 2006.

[8] J. Tang, G. Xue, and W. Zhang, Interference-Aware Topology Control and QoS Routing in Multi-Channel Wireless Mesh Networks, *International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2004.

[9] S. Haykin, Cognitive Radio: Brain-Empowered Wireless Communications, *IEEE Journal on Selected Areas in Communications*, Vol. 23, No. 02, 2005.

[10] http://en.wikipedia.org/wiki/Radio.

[11] D. Grandblaise, D. Bourse, and K. Moessner, Dynamic Spectrum Allocation (DSA) and Reconfigurability, *SDR Forum Technical*, 2002.

[12] P. Kyasanur and N.H.Vaidya, Routing and Interface Assignment in Multi-Channel Multi-Interface Wireless Networks, *UIUC Technical Report*, 2004.

[13] R. Ashish, G. Kartik, and C. Tzicker, Centralized Channel Assignment and Routing Algorithms for Multi-Channel Wireless Mesh Networks, *Mobile Computing and Communications Review*, 2005.

[14] A. Adya, A Multi- Radio Unification Protocol for IEEE 802.11 Wireless Networks, *Broadnets*, San Jose, California, USA, October 2004.

[15] P. Bahl, R. Chandra, and J. Dunagan, SSCH: Slotted Seeded Channel Hopping for Capacity Improvement in IEEE 802.11 Ad-Hoc Wireless Networks, *ACM MobiCom*, Philadelphia, PA, 2004.

[16] L. Cao and H. Zheng, Distributed Spectrum Allocation via Local Bargaining, *Sensor and Ad Hoc Communications and Networks*, 2005.

[17] H. Zheng and C. Peng, Collaboration and Fairness in Opportunistic Spectrum Access, *IEEE Journal on Selected Areas in Communications*, 2005.

[18] M. Marina and S. Das, A Topology Control Approach for Utilizing Multiple Channels in Multi-Radio Wireless Mesh Networks, *Broadnets Symposium*, 2005.

[19] Y.B. Ko and N. H. Vaidya., Location-Aided Routing (LAR) in Mobile Ad Hoc Networks, *ACM/IEEE MobiCom*, 1998.

[20] J. Li and J. Jannotti and D. DeCouto and D. Karger and R. Morris, A Scalable Location Service for Geographic Ad-Hoc Routing, *6th Annu. ACM/IEEE International Conference on Mobile Computing and Networking*, 2000.

[21] S. Basagni and I. Chlamtac and V. R. Syrotiuk and B. A. Woodward, A Distance Routing Effect Algorithm for Mobility (DREAM), *ACM/IEEE MobiCom*, 1998.

[22] F. Kuhn and R. Wattenhofer and A. Zollinger, Asymptotically Optimal Geometric Mobile Ad-Hoc Routing, *6th international workshop on Discrete algorithms and methods for mobile computing and communications*, 2002.

[23] B Leong, S. Mitra and L. Barbara, Path Vector Face Routing: Geographic Routing with Local Face Information, *6th 13th IEEE International Conference on Network Protocols*, 2005.

[24] W. Yan and W. Wu, Data Structure, Section Edition, *Tsinghua University Press*, 2000.

[25] A. Nadeem, K. Salil and J. Sanjay, The Holes Problem in Wireless Sensor Networks: a Survey, *SIGMOBILE Mobile Computing and Communications Review*, Vol.9, 2005.

[26] B. Karp and H. Kung, GPSR: Greedy Perimeter Stateless Routing for Wireless Networks, *ACM/IEEE International Conference on Mobile Computing and Networking*, 2000.

[27] Y.J. Kim, R. Govindan, B. Karp and S. Shenker, Geographic Routing Made Practical, *NSDI*, 2005.

[28] M. Lewis, F. Templin, B. Bellur and R. Ogier, Topology Broadcast Based on Reverse-Path Forwarding, *IETF MANET Internet Draft*, 2002.

[29] E. Kranakis, H. Singh and J. Urrutia, Compass Routing on Geometric Networks, *11-th Canadian Conference on Computational Geometry*, 1999.

[30] V. D. Park and M. S. Corson, A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks, *IEEE INFOCOM*, 1997.

[31] Q. Fang, J. Gao, and L. Guibas, Locating and Bypassing Holes in Sensor Networks, *Mobile Networks and Applications*, No. 11, 2006.

[32] Z. Jiang, J. Ma, W. Lou and J. Wu, An Information Model for Geographic Greedy Forwarding in Wireless Ad-Hoc Sensor Networks, *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, 2008.

[33] C. Liu and J. Wu, Destination-Region-Based Local Minimum Aware Geometric Routing, *IEEE International Conference on Mobile Ad-hoc and Sensor Systems*, 2007.

[34] F. Xing, Y. Xu, M. Zhao and K. Harfoush, HAGR: Hole Aware Greedy Routing for Geometric Ad Hoc Networks, *IEEE/AFCEA Military Communications Conference*, 2007.

[35] B. Raman, Channel Allocation in 802.11-based Mesh Networks, *IEEE Infocom*, 2006.

[36] J. Na and C. K. Kim, GLR: A Novel Geographic Routing Scheme for Large Wireless Ad-Hoc Networks, *Computer Networks*, Vol. 50, 2006.

[37] J. Yang and Z. Fei, ITGR: Intermediate Target Based Geographic Routing, *19th International Conference on Computer Communication Networks*, Zurich, Switzerland, 2010.

[38] J. Yang and Z. Fei, HDAR: Hole Detection and Adaptive Geographic Routing for Ad Hoc Networks, *19th International Conference on Computer Communication Networks*, Zurich, Switzerland, 2010.

[39] B. Karp and H. Kung, GPSR: Greedy Perimeter Stateless Routing for Wireless Networks, *ACM/IEEE International Conference on Mobile Computing and Networking*, 2000.

[40] F. Kuhn and R. Wattenhofer and A. Zollinger, Asymptotically Optimal Geometric Mobile Ad-Hoc Routing, *6th international workshop on discrete algorithms and methods for mobile computing and communications*, 2002.

[41] B. Leong, S. Mitra and B. Liskov, Path Vector Face Routing:Geographic Routing with Local Face Information, *13th IEEE International Conference on Network Protocols*, Boston, MA, 2005.

[42] N. Ahmed, S. Kanhere, and S. Jha, The Holes Problem in Wireless Sensor Networks: a Survey, *SIGMOBILE Mobile Computing and Communications Review*, Vol.9, P4-18, April, 2005.

[43] P. Bose, P. Morin, and I. Stojmenovic, Routing with Guaranteed Delivery in Ad Hoc Wireless Networks, *3rd International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, 1999.

[44] F. Kuhn, R. Wattenhofer, and A. Zollinger, Worst-Case Optimal and Average-Case Efficient Geometric Ad-Hoc Routing, *4th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, 2003.

[45] F. Xing, Y. Xu, M. Zhao, and K. Khaled, HAGR: Hole Aware Greedy Routing for Geometric Ad Hoc Networks, *Military Communications Conference*, 2007.

[46] R. Kleinberg, Geographic Routing Using Hyperbolic Space, *INFOCOM. 26th IEEE International Conference*, 2007.

[47] C. Perkins and P. Bhagwat, Highly Dynamic Destination Sequenced Distance Vector Routing for Mobile Computers, *ACM SIGCOMM*, 1994.

[48] P. Jacquet, P. Muhlethaler and A. Qayyam, Optimized Link-State Routing Protocol, *IETF MANET Internet Draft*, March, 2002.

[49] E. Royer and C. Perkins, Ad-hoc On-Demand Distance Vector Routing, *2nd IEEE Workshop on Mobile Computing Systems and Applications*, 1999.

[50] A. Amoroso, L. Gandolfi, S. Grassilli, and M. Roccetti, FROV: a distributed broadcast protocol for VANET experimental results, *Ultra Modern Telecommunications & Workshops*, 2009, ICUMT.

[51] D. Johnson, D. Maltz, Y Hu and J Jetcheva, The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks, *IETF MANET Internet Draft*, February, 2002.

[52] O.K. Tonguz, N. Wisitpongphan, J.S. Parikh, B. Fan, P. Mudalige, and V.K. Sadekar, On the Broadcast Storm Problem in Ad hoc Wireless Networks, *Broadband Communications, Networks and Systems*, 2006.

[53] V. D. Park and M. S. Corson, A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks, *IEEE INFOCOM*, Japan, 1997.

[54] S.Basagni, I.Chlamtac, V.R.Syrotiuk and B.A.Woodward, A Distance Routing Effect Algorithm for Mobility (DREAM), *ACM/IEEE MobiCom*, 1998.

[55] T. Hou and V. Li, Transmission Range Control in Multi-Hop Packet Radio Networks, *IEEE Transactions on Communications*, Vol. 34(1), P38-44, 1986.

[56] H. Takagi and L. Kleinrock, Optimal Transmission Ranges for Randomly Distributed Packet Radio Terminals, *IEEE Transactions on Communications*, Vol. 32(3), P246-257, 1984.

[57] E. Kranakis, H. Singh and J. Urrutia, Compass Routing on Geometric Networks, *11-th Canadian Conference on Computational Geometry*, 1999.

[58] N. Ahmed, S. Kanhere, and S. Jha, The Holes Problem in Wireless Sensor Networks: a Survey, *SIGMOBILE Mobile Computing and Communications Review*, Vol. 9, April, 2005.

[59] J. Yang and Z. Fei, A New Channel Assignment Algorithm for Wireless Mesh Network, *International Journal of Wireless and Mobile Computing*, Vol. 4, No.1, January, 2009.

[60] GloMoSim, http://pcl.cs.ucla.edu/projects/glomosim/

[61] S. Subramanian, S. Shakkottai and P. Gupta, Optimal Geographic Routing for Wireless Networks with Near-Arbitrary Holes and Traffic, *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, 2008.

[62] Z. Gao, Z. Wu, R. Ye and Q Wu, ICAR Hybrid Network: A Smart In-vehicle Network for Scud-Ware, *The 1st Chinese Conference on Harmonious Human Machine Environment (HHME2005)*, Kunming, China, October 26-28, 2005.

[63] A. Stampoulis and Z. Chai, A Survey of Security in Vehicular Networks, Yale University, 2005.

[64] M. Fiore and J. Harri, The Networking Shape of Vehicular Mobility, *9th ACM international symposium on Mobile ad hoc networking and computing*, 2008.

[65] N. Wisitpongphan, F. Bai, P. Mudalige, V. Sadekar and O. Tonguz, Routing in Sparse Vehicular Ad Hoc Wireless Networks, *IEEE Journal on Selected Areas in Communications*, Vol.25, P1538-1556, October, 2007.

[66] F. Bai, N. Sadagopan and A. Helmy, The IMPORTANT Framework for Analyzing the Impact of Mobility on Performance Of Routing Protocols for Ad Hoc Networks, *Elsevier Ad Hoc Networks*, Vol.1, P383-403, 2003.

[67] J. Yang and Z. Fei, Statistical Filtering Based Broadcast Protocol for Vehicular Networks, *20th International Conference on Computer Communication Networks*, Maui, Hawaii, 2011.

[68] P. Cheng, K. C. Lee, J. Haerri, and M. Gerla, GeoDTN+Nav: Geographic DTN Routing with Navigator Prediction for Urban Vehicular Environments, *Mobile Networks and Applications*, 2009.

[69] S. Jain, K. Fall, and R. Patra, Routing in a Delay Tolerant Network, *SIGCOMM*, 2004.

[70] J. Ott and D. Kutscher, A Disconnection-Tolerant Transport for Drive-Thru Internet Environments, *INFOCOM*, 2005.

[71] M. Musolesi, S. Hailes, and C. Mascolo, Adaptive Routing for Intermittently Connected Mobile Ad Hoc Networks, *Sixth IEEE International Symposium on world of wireless mobile and multimedia networks*, 2005.

[72] I. Leontiadis and C. Mascolo, GeOpps: Geographical Opportunistic Routing for Vehicular Networks, *World of wireless, mobile and multimedia networks*, 2007.

[73] D. Nain, N. Petigara, H. Balakrishnan, Integrated Routing and Storage for Messaging Applications in Mobile Ad Hoc Networks, *Mob Netw Appl*, 2004.

[74] K. Lee, P. Cheng, J. Weng, L. Tung, and M. Gerla, VCLCR: a Practical Geographic Routing Protocol in Urban Scenarios, *Technical report 080009, UCLA*, March 2008.

[75] S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan, and S. Shenker, GHT: A Geographic Hash Table for Data-Centric Storage in Sensornets, *1st ACM International Workshop on Wireless Sensor Networks and Applications (WSNA)*, 2002.

Vita

Personal Data:

Name: Jianjun Yang

Gender: Male

Educational Background:

- Masters of Science in Computer Science, University of Kentucky, 2010.
- Masters of Science in Computer Science and Technology, Fuzhou University, 1999.
- Bachelor of Science in Computer Science, Fuzhou University, 1996.

Research Experience:

- Ph.D. Research Field: Computer Networks, 2006 - Present.
  The Laboratory for Advanced Networking,
  Department of Computer Science, University of Kentucky, Lexington, KY, USA.

- Master Research Topic: Geographic Information Systems, 1996-1999.
  National Laboratory of Geographic Information Systems, Fuzhou, China.

Teaching Experience:

- Teaching Assistant, 01/2006 - 05/2011.
  Department of Computer Science, University of Kentucky, Lexington, KY, USA.

Work Experience:

- Network Administrator and Computer Engineer, 03/1999 - 08/2005.
  Shenzhen Water Group Company, Shenzhen, China.

Awards:

- Winner of the student travel grant, 19th International Conference on Computer Communications and Networks, Zurich, Switzerland, 2010.
- Winner of the student travel grant, 6th USENIX Conference on File and Storage Technologies, San Jose, CA, 2008.
- "The Teaching Assistant with the Fastest Red Pen," University of Kentucky, 2007.
- Excellent project award of "The algorithm to convert dynamic web pages to static web pages," Science Conference in Shenzhen, Shenzhen, China, 2004.
- Outstanding employee, Shenzhen Water Group Company, Shenzhen, China, 2003 and 2004.
- Outstanding Student Fellowship, Fuzhou University, China, 1998.
- Award of "Most Outstanding Undergraduate," Fuzhou University, China, 1996.

Publications:

- Jianjun Yang and Zongming Fei, "Statistical Filtering Based Broadcast Protocol for Vehicular Networks," *20th International Conference on Computer Communication Networks*, Maui, Hawaii, 2011.

- Jianjun Yang and Zongming Fei, ITGR: Intermediate Target Based Geographic Routing, *19th International Conference on Computer Communication Networks*, Zurich, Switzerland, 2010.

- Jianjun Yang and Zongming Fei, HDAR: Hole Detection and Adaptive Geographic Routing for Ad Hoc Networks, *19th International Conference on Computer Communication Networks*, Zurich, Switzerland, 2010.

- Jianjun Yang and Zongming Fei, A New Channel Assignment Algorithm for Wireless Mesh Networks, *International Journal of Pervasive Computing and Communication*, Vol. 5, No. 3, 2009.

- Jianjun Yang and Zongming Fei, Bipartite Graph Based Dynamic Spectrum Allocation for Wireless Mesh Networks, *2nd IEEE International Workshop on Wireless Mesh and Ad Hoc Networks* , Beijing, 2008.

- Jianjun Yang, Development of Web Publishing System Based on PHP and SQL Server, *1st International Information Construction Workshop in Hong Kong, Macao, and Shenzhen*, Hong Kong, 2005.

- Jianjun Yang and Lei Tang, An Information Search Algorithm Based on Binary Tree in Internet, *Automation Engineering Technology*, Vol. 9, No. 6, 2004.

- Jianjun Yang, An Optimal Quick Algorithm for Water Charge Query System, *Shenzhen Water Automation Engineering Technology* , Vol. 3, 2003.