DISSERTATION

Advantages and Limitations of Position-based Communication in Wireless Ad-hoc Networks

ausgeführt zum Zwecke der Erlangung des akademischen Grades
eines Doktors der technischen Wissenschaften
unter der Leitung von

UNIV.PROF. DR. ULRICH SCHMID

Inst.-Nr. E182/2
Institut für Technische Informatik
Embedded Computing Systems Group

eingereicht an der Technischen Universität Wien
Fakultät für Informatik

von

HANNES STRATIL
Matr.-Nr. 9625427

Grazerstraße 85a
8605 Kapfenberg
Austria

Wien, 17. Februar 2006 _____

To my mother
Helga Stratil
June 9, 1943 – October 5, 2005

Seid gewiss: Ich bin bei euch alle
Tage bis zum Ende der Welt.

Mt 28, 20

Acknowledgments

I am grateful to my adviser Prof. Ulrich Schmid for his patience and experienced supervision on this thesis. I am also thankful to Prof. Roger Wattenhofer for his kind acceptance to serve as second adviser and committee member.

I would like to express my sincere thanks to my colleagues at the Vienna University of Technology for inspiring discussions – both regarding scientific and more common subjects – and a great working environment.

This research was financially supported by the Austrian Academy of Sciences through through a DOC scholarship. This support is gratefully acknowledged.

Vienna, February 17, 2006

Hannes Stratil

Abstract

Wireless ad-hoc networks are a hot topic in distributed computing research, which is fueled by promising future applications of military, government and commercial customers. Wireless ad-hoc networks are made up of computing nodes that communicate with each other over a wireless medium in the absence of a fixed infrastructure and any centralized control. Direct communication between arbitrary nodes is generally not possible and requires a multi-hop routing protocol.

This thesis presents SDT/VAR, an efficient protocol for communication in wireless ad-hoc networks. The Short delaunay triangulation (SDT) is a powerful topology based on the construction rules of the Delaunay triangulation. The Voronoi-aided routing (VAR) protocol is an efficient implementation of the well known greedy/perimeter routing approach atop of the SDT. The big advantage of our approach is efficiency: The computation of the Short delaunay triangulation automatically provides the local Voronoi diagrams required for the efficient implementation of Voronoi-aided routing. Algorithms proposed until today generally separate topology control and routing. Our findings reveal that a joint development based upon a common efficient data structure leads to considerably increased performance.

The second important part of this thesis is on fault-tolerance and reliability of greedy/perimeter routing. Most existing wireless ad-hoc network routing protocols assume a fault-free environment during message delivery. However, this assumption does not hold in realistic environments. We analyze the behavior of greedy/perimeter routing in the context of crash failures and the reliability of greedy/perimeter routing with respect to imprecise position information and mobile nodes.

Some results of this thesis have been published previously. Chapter 6 is based on [Str05a], parts of Chapter 8 are based on [Str04b], and Chapter 10 is based on [Str05b].

This research is supported by the OEAW — Austrian Academy of Sciences — through a DOC scholarship, by the Austrian Science Fund (FWF) under grant no. P18264-N04 (SPAWN-Project) and by the Austrian START programme (no. Y41-MAT).

Zusammenfassung

Der Begriff *Wireless ad-hoc Network* lässt sich ungefähr mit *drahtloses, infrastrukturloses Netzwerk* übersetzen und bezeichnet eine neue Form von drahtlosen Netzwerken. Die Teilnehmer (Stationen) in einem solchen Netzwerk können drahtlos (meistens per Funk) miteinander kommunizieren und benötigen keine vorgegebene Infrastruktur oder zentrale Kontrollinstanz. Im allgemeinen ist es nicht möglich, daß jeder Teilnehmer mit jeden anderen Teilnehmer direkt kommuniziert. Ein Ad-hoc-Netzwerk benötigt für die Datenübertragung zwischen beliebigen Teilnehmern einen oder mehrere Hops, also Teilstrecken zwischen individuellen Stationen. Multi-hop-Systeme werden häufig als Mobilfunksysteme der vierten Generation bezeichnet.

Die vorliegende Dissertation präsentiert mit dem SDT/VAR-Protokoll ein effizientes Kommunikationsprotokoll für die Datenübertragung in drahtlosen Ad-hoc-Netzwerken. Dieses Protokoll verwendet die Prinzipien der Delaunay-Triangulation für den Aufbau einer effizienten Netzwerktopologie und das duale Voronoi-Diagramm für die Implementierung von Greedy/Perimeter-Routing.

Der zweite zentrale Teil dieser Dissertation beschäftigt sich mit der Fehlertoleranz von Greedy/Perimeter-Routing. Wir untersuchen das Verhalten von Greedy/Perimeter-Routing in einem fehleranfälligen Netzwerk und ermitteln den Aufwand, der notwendig ist, um die Datenübertragung in einem Netzwerk mit Crash-Fehlern zu garantieren. Weiters analysieren wir die Zuverlässigkeit von positionsbasierenden Protokollen (wie Greedy/Perimeter-Routing) wenn keine exakten Positionsangaben verfügbar sind und zeigen, daßPerimeter-Routing die Kommunikation nicht garantieren kann, wenn die Teilnehmer im Netzwerk mobil sind.

Teile dieser Dissertation wurden schon im Vorfeld auf verschiedenen Konferenzen vorgestellt. Kapitel 6 basiert auf [Str05a], Teile von Kapitel 8 wurden bereits in [Str04b] publiziert und Kapitel 10 fußt auf [Str05b].

Diese Dissertation wurde durch ein DOC-Stipendium der Österreichischen Akademie der Wissenschaften sowie vom Fonds zur Förderung der wissenschaftlichen Forschung (FWF) durch das START Programm (Nr. Y41-MAT) und das Projekt Nr. P18264-N04 (SPAWN) unterstützt.

Contents

Alles was ein Mensch sich
vorstellen kann, werden andere
Menschen verwirklichen.

Jules Verne (1828–1905)

Chapter 1

Introduction

A *wireless ad-hoc network* consists of a collection of communication nodes. The nodes are randomly dispersed over some area of interest and communicate with each other over a wireless medium in the absence of a fixed infrastructure and any centralized control. Any computation in a wireless ad-hoc network should be decentralized. Hence, distributed algorithms and protocols are required. Efficient and scalable algorithms for wireless ad-hoc networks may only use information about the local neighborhood. We call this *localized*. A survey of wireless ad-hoc networks and other well-known wireless technologies is given in Chapter 2.

We assume the existence of a *position service*, which provides network participants with their location. The Global Position Service (GPS) is definitely the best known position service in use today. Position information enables context-sensitive computing and is the basis for many location specific applications, such as service discovery, resource discovery and mapping. Location knowledge is also an important information for supervision and security features.

A wireless ad-hoc network requires a *routing protocol* for the communication in the network. Routing in wireless ad-hoc networks is nontrivial since mobility, erroneous nodes and changes in node activity status cause frequent and unpredictable topology changes. Routing is usually addressed at the network layer of the OSI 7-layer hierarchy. The overall network layer responsibilities are divided between the topology control sublayer and the routing sublayer. *Topology control* and routing are two of the common problems in the context of wireless ad-hoc networks. At the topology control sublayer, the network can be simply model as a *communication graph* and each edge is an independent non-interfering link. The communication graph of a wireless ad-hoc network can be seen in Figure 1.1. The aim of the topology control sublayer is to construct an appropriate topology to improve the efficiency and performance of the overlying algorithms. The rout-

ing sublayer manages the message exchange between non-neighboring nodes in a wireless ad-hoc network.

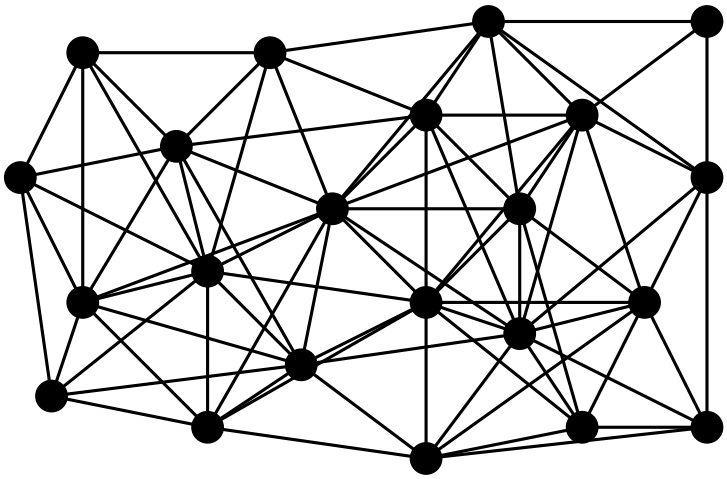


Figure 1.1: **Communication Graph:** *Two nodes have a communication edge, if the distance between them is shorter than the communication range.*

The topology plays a key role in the performance of the routing algorithm. The techniques to compute an appropriate topology are called *topology control* and form an important part of research in wireless ad-hoc networking. Each node can normally communicate with a lot of other nodes, the wireless ad-hoc network is perhaps fully connected, i.e., each node can communicate with all other nodes in the network. The task of topology control is a reasonable restriction of the available communication links to a small number of beneficial links. An important parameter for the restriction of the communication links is energy efficiency. A sequence of short communication links requires less power than one long communication link and the susceptibility to communication faults increases with increasing distance. The limitation of data exchange to the nearest neighbors reduces also the interference level in the network, because exclusive communication in the local neighborhood has no influence to nodes further away. Hence, the throughput of the network will be increased. However, the communication links must be carefully chosen, because connectivity is still the most important requirement for a topology. Chapter 3 gives an overview of topology control. The most important quality properties will be presented and the different approaches are shown.

A very interesting class of topology control algorithms are position-based ap-

proaches. These algorithms use the construction rules of *proximity graphs* for the computation of the topology. Proximity graphs belong to the most marvelous structures in computational geometry. These graphs offer an almost magical connectivity between the elements in the graphs. Some interesting properties of proximity graphs and the relations between these graphs are presented in Chapter 4. A topology based on a proximity graph inherits the properties of the chosen proximity graph. However, proximity graphs are not directly applicable to position-based topology control. Computation in wireless ad-hoc networks must be done totally localized and communication links are restricted by the maximal communication range, whereas proximity graphs are computed centralized by one process and edges can be arbitrarily long. These issues are addressed in Chapter 5.

A well-known proximity graph is the *Delaunay triangulation*. A lot of properties which are attractive for topologies are concentrated on it. The Delaunay triangulation is sparse, connected, planar, and has a constant length stretch factor. However, the dissemination of these properties to a topology based on the Delaunay triangulation requires some effort. The *Short delaunay triangulation* (SDT) protocol presented in Chapter 6 provides these properties and computes an efficient topology for wireless ad-hoc networks. The algorithm computes the topology totally localized. This guarantees scalability, because the algorithm is independent of the total number of nodes in the wireless network. Figure 1.2 shows the Short delaunay triangulation computed from the communication graph of Figure 1.1.

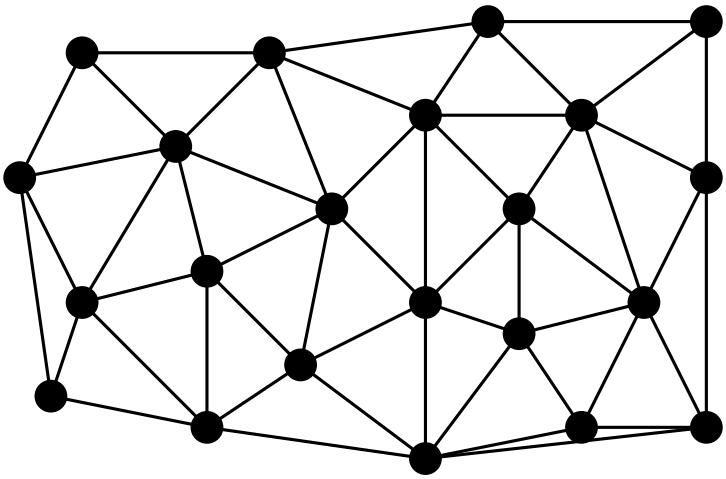


Figure 1.2: **Short Delaunay Triangulation:** *Contains all edges of the Delaunay triangulation that are shorter than the communication range.*

A wireless ad-hoc network requires a *multi-hop routing protocol* for the communication between non-neighboring nodes, since direct communication between arbitrary nodes is usually impossible. Chapter 7 presents a taxonomy of common routing algorithms. A very promising category of algorithms for the communication in wireless ad-hoc networks are position-based protocols. A well-known example for a reliable position-based routing algorithm is the *greedy/perimeter routing* approach.

The greedy forwarding approach is a totally localized approach: The routing decision at a node is only based on its own position, the position of its single hop neighbor nodes and the position of the destination node. Greedy routing does not require the establishment or maintenance of routes: The nodes neither have to store routing tables nor do they transmit messages to keep the routing tables up-to-date, and no global information about the topology of the network is needed. Greedy routing requires neither flooding nor the distribution of status information to further nodes than the single hop neighbor nodes. When an intermediate node receives a message for a specific destination node, it forwards a message to that neighbor node who is closest to the destination node, i.e., the neighbor node with the shortest euclidean distance to the destination node.

The question of how to find the closest site according to a given query point is one of the best known problems in the context of *Voronoi diagram*s, called *point location* or the *post-office problem*. Voronoi diagrams offer an efficient solution for point location: A Voronoi diagram divides the plane into *Voronoi cell*s and each cell contains all points closer to the site of this cell than to all other sites. To find the site closest to a point, it suffices to determine the Voronoi cell that contains the point.

For greedy routing, a node uses its own position and the positions of its neighbor nodes to compute a local Voronoi diagram. When a node wants to forward a message to a destination node, it determines the cell the destination node belongs to (using the point-location algorithm), and forwards the message to that unique neighbor node that is the site of this cell. The implementation of greedy routing with Voronoi diagrams is called *Voronoi-aided routing* (VAR) and is presented in Chapter 8. Figure 1.3 shows an example of Voronoi-aided routing.

Greedy routing can be used until the message reaches the destination or a node where no neighbor is closer to the destination than the node itself. At such a local minimum, greedy routing is no longer possible and a recovery strategy is required. A well-known recovery strategy is perimeter routing which forwards the message along the faces of a planar subgraph. If a node is reached, with position is closer

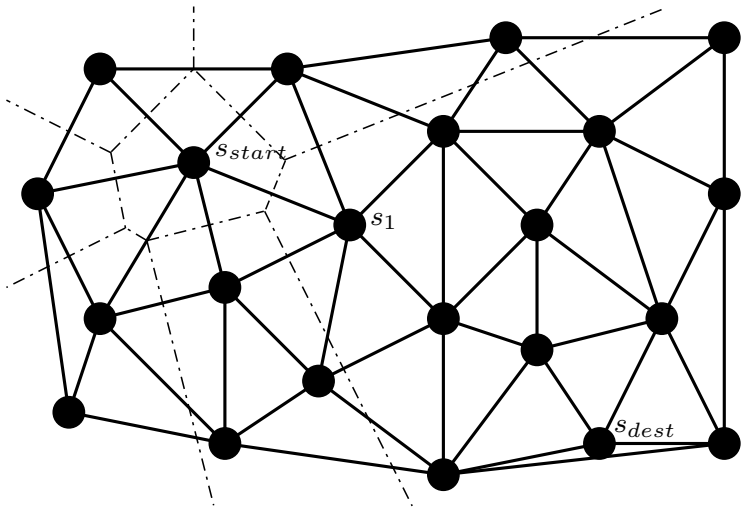


Figure 1.3: **Greedy Routing:** *Node s_{start} wants to send a message to node s_{dest}. Node s_{start} use its local Voronoi diagram and a point-location algorithm to find the neighbor node which is closest to the destination — this is node s_1 — and forwards the message to this node. Each node to the same until the message reach the destination.*

to the destination node than the node where the greedy strategy previously failed, the greedy routing algorithm takes over control again. An example of greedy/ perimeter routing is shown in Figure 1.4.

The planarity of the topology is an indispensable property to guarantee the functionality of perimeter routing. The Short delaunay triangulation is not only an efficient topology but also planar. Each node can use its local contribution to the Short delaunay triangulation for perimeter routing and the dual of its local contribution, the local Voronoi diagram, for greedy routing. Hence, the computation of the topology yields automatically to an efficient data-structure for greedy/ perimeter routing.

Fault-tolerance is blatantly ignored in the research and development of topology control and routing algorithms until today. Hence, the second part of this thesis is on the fault tolerance of position-based communication.

Erroneous or malicious nodes can destroy the planarity and the connectivity of a topology. Unreliable routing approaches compromise the functionality of the wireless ad-hoc network and decrease the throughput. Chapter 9 presents different approaches to improve the fault-tolerance of greedy/perimeter routing.

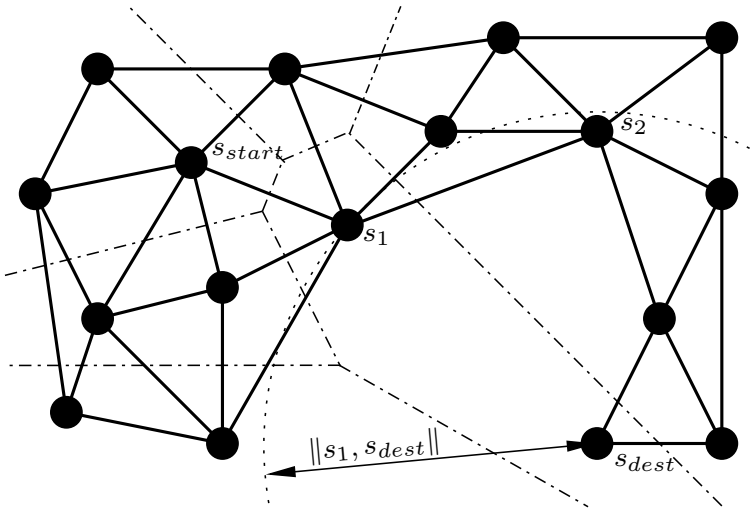


Figure 1.4: **Greedy/Perimeter Routing:** *We assume for this example an empty region in the network. Node s_{start} starts a message to s_{dest} and forwards the message in greedy mode to node s_1. Node s_1 is the closest node to the destination node, because s_{dest} lies in the Voronoi Cell of s_1. Hence, s_1 initiates the perimeter mode and forwards the message to node s_2. Node s_2 is closer to s_{dest} than node s_1 and so the message will be forwarded in greedy mode again.*

The most sensitive part of greedy/perimeter routing is the required planarity of the topology. The computation of the topology must be done localized. Each node computes only its own contribution to the overall topology. The status of a node (correct of faulty) must agree on all neighbor nodes. Inconsistencies can yield to a loss of planarity which can have bad implication on routing. In particular, a message can get into a loop. The aim of Chapter 10 is to analyze the requirements that are necessary to tolerate crash failures of nodes in the topology and to facilitate a reliable routing algorithm the delivery of messages.

Faulty nodes are not the only menace to planarity of topologies. Essential for the computation of a planar topology are furthermore the positions of the nodes and the availability of sufficient information for the localized computation. Chapter 11 specifies the requirements on the communication range that are necessary for correct and localized computation of wireless ad-hoc network topologies. However, all position services existing today have a certain degree of inaccuracy. The estimated positions differ from the real positions and the topology control algorithm must consider the discrepancies. Also the distribution of position information among the neighbor nodes influences the construction of a

position-based topology. Common topology control algorithms assume that all nodes obtain changes in node positions immediately, but the node positions can be inconsistent on neighbor nodes for a certain amount of time. Chapter 12 extends the requirements specified in Chapter 11 to tolerate imprecise position information and investigates the robustness of a planar topology against inconsistent position information and node movement. At the end of Chapter 12, we show in an example that it is impossible to ensure the planarity of the topology in the presence of mobile and new joining nodes. Hence, the simple greedy/perimeter routing approach is unable to guarantee message delivery.

The final Chapter 13 summarize the results of the previous chapters and discusses some possible applications for the VAR/SDT protocol. An overview of different algorithms for the computation of the Delaunay triangulation and an investigation of the usability of these algorithms in dynamic environments is given in Appendix A.

Ordnung ist etwas Natürliches. Das Nützliche ist das Chaos.

Arthur Schnitzler (1862–1931)

Chapter 2

Wireless ad-hoc Networks

2.1 Wireless Networks

Wireless networking has emerged as its own discipline over the past decade. From cellular voice telephony to wireless access to the Internet and wireless home networking, wireless networks have profoundly influenced our lifestyle. After a decade of exponential growth, today's wireless industry is one of the largest industries in the world.

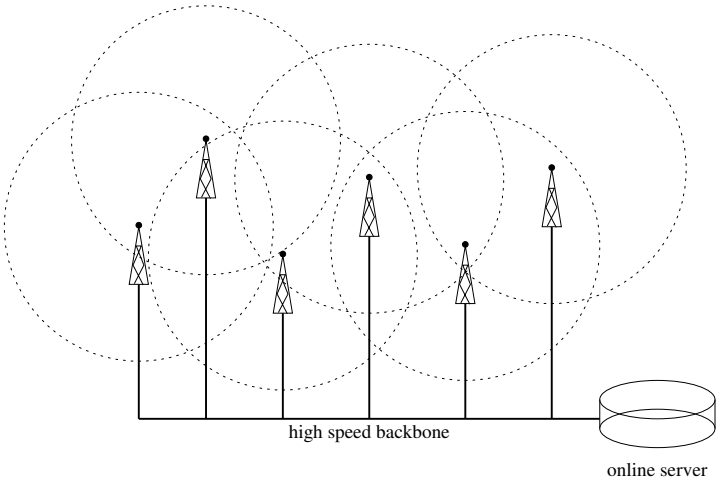


Figure 2.1: *Cellular Network*

Today's wireless networks consist of a fixed (wired) infrastructure that supports communication between mobile terminals and between mobile and fixed terminals. Figure 2.1 shows an example of a wireless network with a fixed wired backbone. Network topologies are often designed for large coverage areas and multiple base station or access point operations. The base station serves as a hub

of the network, and the mobile terminals are located at the ends of the spokes. Any communication from one wireless user station to another has to be sent through the base station. The base station usually controls the mobile stations and monitors what each station is transmitting. Well known examples of wireless networks are wireless local area network (WLAN) and wireless wide area network (WWAN). A wireless local area network generally covers a premise of an organization such as a business corporation or a university. Products based on the IEEE 802.11 [Dep97] standards have become wildly popular. A wireless wide area network covers the geographic space of a city, a country or even larger. Cellular telephony is the most famous example of a wireless technology. The wireless wide area networks used in cellular telephony are also called cellular networks. Worldwide, over two billion people use mobile phones.

Wireless personal area networks (WPAN) are another well known class of wireless ad-hoc networks. A wireless personal area network is a network constituted by connected devices placed inside a circle of 10 meters. Bluetooth is a very successful implementation of the WPAN technology. The Bluetooth specifications [Blu] are released by the Bluetooth Special Interest Group. The personal area networking technology is now standardized by the IEEE 802.15 standards [Dep02].

More recently wireless networking research has shifted towards ad-hoc networks. The idea of ad-hoc or distributed networks has been under development from 1970s in the framework of Mobile Packet Radio technology. In the middle of the nineties, with the definition of standards (e.g., IEEE 802.11), commercial radio technologies have begun to appear and the wireless research community identified in wireless ad-hoc networks a formidable challenging evolution of wireless networks. Wireless ad-hoc networks can operate in the absence of a fixed infrastructure and any centralized control. Any computation in a wireless ad-hoc network hence needs to be carried out in a decentralized manner and should be localized (i.e., use information of its local neighborhood only), self-organized and self-stabilizing. Wireless ad-hoc networks are a collection of nodes that are randomly dispersed over some area of interest. Each node in a wireless ad-hoc network functions as both a host and a router, and the control of the network is distributed among the nodes. The network topology is in general dynamic, because the connectivity among the nodes may vary with time due to node departures, new node arrivals, and the possibility of having mobile nodes. Hence, there is a need for efficient routing protocols to allow the nodes to communicate possibly over multi-hop paths consisting of several links in a way that does not use any more of the network resources than necessary. Some features of wireless ad-hoc networks were studied since the 1970s and others are motivated by the increasing number

of applications. Anyway, research in the area of wireless ad-hoc networking is receiving much attention from academia, industry, and government.

The following enumeration illustrates some of the main advantages of ad-hoc networks over traditional wireless networks:

Scalability. In single hop networks, expansion is always limited to the coverage of the radio transmitter and receiver, and there is no simple way to scale up the network coverage. In ad-hoc networks, as the number of participants increases the potential coverage of the network is increased.

Flexibility. Construction of a traditional wireless network requires deployment of a network infrastructure which is very often time and money consuming. Ad-hoc networks are inherently flexible and can be set up instantly.

Reliability. Another issue of concern in wireless applications is resistance to partial failure. Traditional networks are "single point of failure" networks. If a base station fails, the entire communication network is destroyed. This problem does not exists in ad-hoc networks.

2.2 Wireless ad-hoc Networks

Since the advent of DARPA packet radio networks in the early 1970s [JT87], wireless ad-hoc networks have become an interesting research object in science and in computer industry. Every node in a wireless ad-hoc network is a device that integrates at least a processing unit, some memory, a power unit and a wireless communication interface. The communication interface is used to transmit messages to neighbor nodes and to receive messages from neighbor nodes. The communication in wireless ad-hoc networks is usually asynchronous, because the nodes utilize no common time base. Since the nodes communicate over wireless links, they have to contend with the effects of radio communication, such as noise, fading, and interference. In addition, the links typically have less bandwidth than in a cellular network. Wireless ad-hoc networks are considered useful for a wide range of application domains including military applications (establishing communication among a group of soldiers for tactical operations), inventory tracking, surveillance, reconnaissance, targeting systems, environment monitoring (fire or flood detection), and wireless sensor networks.

Common examples of wireless ad-hoc networks are mobile ad-hoc networks, mesh networks, and sensor networks. Parts of the following text are taken from the book *Ad Hoc Wireless Networks* by C. Siva Ram Murthy and B.S. Manjo [MM04].

2.2.1 Mobile ad-hoc Networks (MANETs)

A mobile ad-hoc network (Figure 2.2) is a collection of wireless nodes that can dynamically be set up anywhere and anytime. The nodes are mobile and free to move randomly. In mobile ad-hoc networks is often a need for the rapid deployment of independent mobile users. Significant examples include establishing survivable, efficient, dynamic communication for emergency/rescue operations, disaster relief efforts, and military networks. Such network scenarios cannot rely on centralized and organized connectivity. A MANET is an autonomous collection of mobile users that communicate over relatively bandwidth constrained wireless links. Since the nodes are mobile, the network topology may change rapidly and unpredictably over time. The network is decentralized, therefore all network activity including discovering the topology and delivering messages must be executed by the nodes themselves, i.e., routing functionality has to be incorporated into mobile nodes.

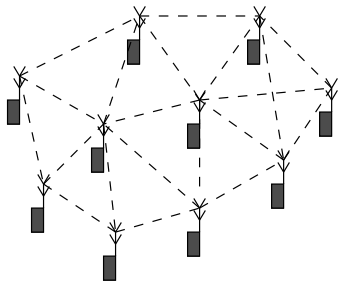


Figure 2.2: *Mobile ad-hoc Network*

The concepts and operational requirements associated with the current idea of MANETs are discussed in the mobile computing and networking literature, notably documents and standards developed by the MANET Working Group of the Routing Area of the Internet Engineering Task Force (IEFT).

2.2.2 Wireless Mesh Networks

Wireless mesh networks (Figure 2.3) are ad-hoc networks that are formed to provide an alternative communication infrastructure for mobile or fixed users, without the spectrum reuse constraints and the requirements of network planning of cellular networks. Wireless mesh networks, or rooftop networks (the name refers to antennas on building's roofs), are not mobile, but are deployed very densely in metropolitan areas as an alternative to wired networking offered by traditional telecommunication providers. Such a network also provides an alternative infrastructure in the event of failure of the conventional one, as after a disaster.

The mesh topology of wireless networks provides many alternate paths for a data transfer session between source and destination, resulting in quick reconfiguration of the path when the existing path fails due to node failures. The investment required in wireless mesh networks is much less than what is required for the cellular network counterparts. Such networks are formed by placing wireless relaying equipment spread across the area to be covered by the network. The possible deployment scenarios of wireless mesh networks include: residential zones (where broadband Internet is required), highways (where a communication facility for automobiles is required), business zones (where an alternate communication system to cellular networks is required), and university campuses (where inexpensive campus-wide network coverage can be provided). Wireless mesh networks should be capable of self-organization and maintenance. The ability of the network to overcome single or multiple node failures resulting from disasters makes it convenient for providing the communication infrastructure for strategic applications.

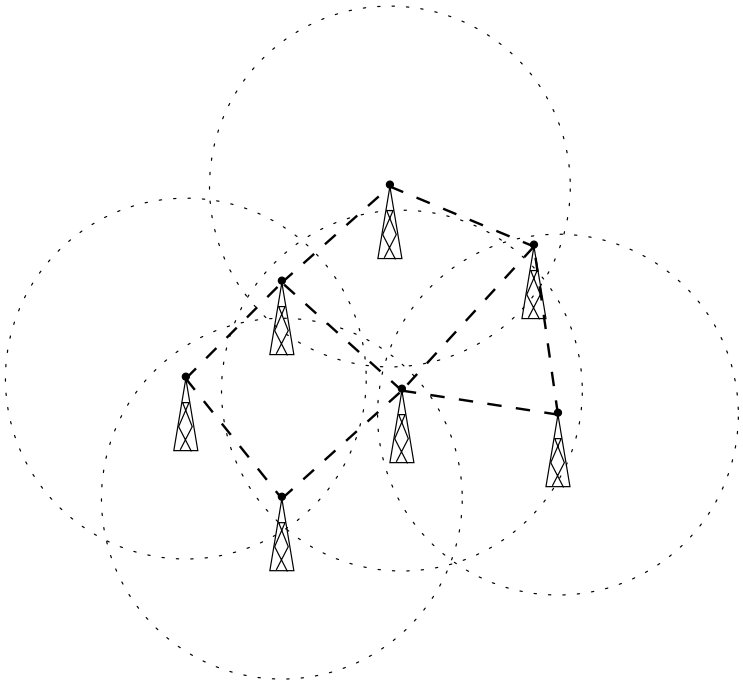


Figure 2.3: *Wireless Mesh Network*

2.2.3 Wireless Sensor Networks

Sensor networks are a special category of wireless ad-hoc networks that are used to provide a wireless communication infrastructure among the sensors deployed

in a specific application domain. Figure 2.4 gives an example of such a network. Sensor nodes are tiny devices that have the capability of sensing physical parameters, processing the data gathered, and communicating the data over the network to the monitoring station. A sensor network is a collection of a large number of sensor nodes that are deployed in a particular region. The activity of sensing can be periodic or sporadic. An example for the periodic type is the sensing of environmental factors such as temperature, humidity, and nuclear radiation. Detecting border intrusion and measuring the stress on critical structures or machinery are examples of the sensing activities that belong to the sporadic type. Some of the domains of application for wireless sensor nodes are military, health care, home security, and environmental monitoring. The issues that make sensor networks a distinct category of wireless ad-hoc networks are the following:

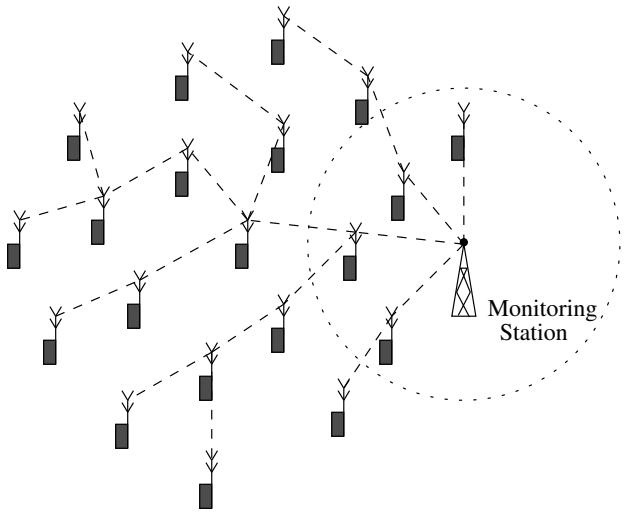


Figure 2.4: *Wireless Sensor Network*

- **Mobility of nodes.** Mobility of nodes is not a mandatory requirement in sensor networks. For example, the nodes deployed for periodic monitoring of soil properties are not required to be mobile. However, the sensor nodes that are fitted on the bodies of patients in a post-surgery ward of a hospital wing may be designed to support limited mobility. Sensor nodes are generally stationary after dissemination.

- **Size of the network.** A sensor network consists of hundreds to thousands of nodes, so the number of nodes is several orders of magnitude higher than the number of nodes in an traditional ad-hoc network.

- **Power constraints.** The power constraints in wireless sensor networks are much more stringent than those in wireless ad-hoc networks. This is mainly because the sensor nodes are expected to operate in harsh environmental or geographical conditions, with minimum or no human supervision and maintenance. In certain cases, the recharging of the energy source is impossible. Running a network, with nodes powered by a battery source with limited energy, demands very efficient protocols at network, data-link, and physical layer.

- **Activity Status.** Sensor nodes change their activity status for reducing energy consumption.

- **Communication.** The traffic in sensor networks is generally reduced to communication from the sensor nodes to a central monitoring station and to broadcasts, multicasts, or geocasts of the monitoring station.

- **Data Fusion.** The limited bandwidth and power constraints demand aggregation of multiple packets into bone at the intermediate relay nodes that are responsible for relaying. This mainly aims at reducing the bandwidth consumed by redundant headers of the packets and reducing the media access delay involved in transmitting multiple packets.

2.3 Communication in Wireless ad-hoc Networks

Communication in wireless ad-hoc network does not require the existence of a central base station or a fixed network infrastructure. Each node of a wireless ad-hoc network can be the destination of some messages while at the same time it can function as relay station for other messages to their final destination. This multi-hop support in wireless ad-hoc networks, which makes communication between nodes outside direct radio range of each other possible, is probably the most distinct difference between wireless ad-hoc networks and wireless LANs.

Each node in a wireless ad-hoc network can be characterized by its computational and communication power. The computational power of a node determines the level of coding and encryption a node is able to perform, two key issues in wireless communication. The communication characteristics of the network are governed by the propagation characteristics of the radio channel, the environment, the battery power, and the power control capabilities of the individual nodes. Ad-hoc networks typically consist of identical nodes, i.e., each node has the same computational and communication power. Radio propagation and interference

models can be used to derive meaningful bounds on the capacity of wireless ad-hoc networks, given node locations and transmission power constraints.

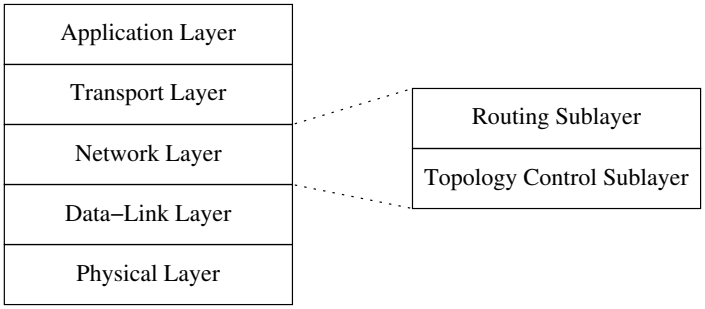


Figure 2.5: *OSI Reference Model*

Communication in a wireless ad-hoc network can be described by the ISO/OSI Reference Model [74984] (Figure 2.5). Power control and interference are usually addressed at the data-link layer of the OSI layer hierarchy. The Medium Access Control (MAC) responsibilities at the data-link layer (e.g., the IEEE 802.11 MAC protocol) creates reliable point-to-point connections between the nodes in the wireless ad-hoc network. This forms the basic infrastructure needed for wireless multi hop communication. Another objective of the MAC protocol is fair and efficient sharing of the communication resources between the nodes. The data link layer provides to the network layer an error-free reliable link between two nodes. The overall network layer responsibilities are divided between the topology control sublayer and the routing sublayer. Thus, at the topology control sublayer, the network can be simply modeled as a communication graph and each edge is an independent non-interfering link. The topology control sublayer computes an appropriately constructed subgraph of the communication graph to improve the performance of the routing algorithm. The routing sublayer manages the message exchange between the nodes in the wireless ad-hoc network. The aim of the transport layer is the end-to-end communication between sender and receiver. The transport layer handles duplicates and end-to-end retransmissions and deals with addressing.

Direct communication between two nodes in a wireless ad-hoc network is only guaranteed if the distance between them is less than the minimum of their respective communication ranges. It is generally not possible (nor desirable) that all nodes are within the communication range of each other. A wireless ad-hoc network uses a multi-hop routing protocol to enable the communication between non-neighboring nodes. Hence, two far-apart nodes in a wireless ad-hoc network

can communicate with each other through the relay of intermediated nodes. The nodes must cooperate to deliver messages across the network and therefore, each node in a wireless ad-hoc network must be aware of the neighboring nodes in its coverage range.

The communication between proximate nodes has to be preferred against the communication between widely separated nodes, because a node needs less transmission power for the communication to a proximate node. This has two advantages: (1) reducing the signal interference and (2) saving transmission power.

ad (1). In wireless ad-hoc networks communication between nodes takes place over radio channels. As long as all nodes use the same frequency band for communication, any node-to-node transmission will add to the level of interference experienced by other users. If nodes communicate with their neighbor nodes only, the interference level in the network will stay low and the overall throughput in the network increases.

ad (2). Nodes in wireless ad-hoc networks are limited in power capacities, because they are normally equipped with batteries. In the most common power-attenuation model, the power to support a link uv is assumed to be $\|uv\|^{\alpha}$, where $\|uv\|$ is the Euclidean distance between u and v, and α is a constant between 2 and 6 depending on the wireless transmission environment. The lifetime of a wireless ad-hoc network can be increased, if the nodes only communicate with the nearest neighbor nodes and not with all nodes.

Multi-hop routing is the most popular research topic in the context of wireless ad-hoc networks. In the last years, hundreds of papers were published about different routing approaches and many more about simulations of these approaches in different environments and under different circumstances. An overview of common routing approaches is presented in Chapter 7. The decision for a specific routing approach depends primary on the degree of mobility and the number of nodes in the wireless ad-hoc network. Every routing algorithm, no matter how absurd or complicated it is, works in a static network with a small number of nodes, and no routing algorithm, no matter how awesome and sophisticated it is, works in a highly dynamic network with a huge number of nodes. Above a specific threshold flooding is the only way to communicate with other nodes.

A node can transmit a message to another node in the network, if it is aware of some global information which gives the node the possibility to recognize its place in the wireless network. This information is either a topological (i.e., the

node knows the connections to the other nodes) one or geographic (i.e., the node knows its geographic position). The nodes must either exchange connectivity information to acquire the topological information or require a position service which provides the participants with their positions. The exchange of connectivity information is limited by the network size whereas position estimation is independent of the number of nodes in the network.

2.4 Position-based Communication

A very promising category of algorithms for communication in wireless ad-hoc networks are position-based protocols. They assume the existence of a position service, which provides network participants with their location , e.g., by triangulation with the help of fixed beacons or the positions of neighbor nodes (e.g., [NN01, SRL02]). The Global Position Service (GPS) is undoubtedly the most well known position service in use today. Unfortunately, GPS is unsuitable for some applications because GPS does not work indoors. An example of an indoor position services is presented in He et al. [HHB+03]. A survey of position services can be found in [HB01].

Position information enables context-sensitive computing and is the basis of many location specific applications, such as service discovery, resource discovery and mapping. Location knowledge is also an important information for supervision and security features. In a wireless ad-hoc network the position is in fact more important than a specific node ID. For example, tracking applications are more interested in where the target is located, as in the ID of the reporting node.

Position information is an auxiliary feature for communication in wireless ad-hoc networks. Position-based routing, or geographic routing, uses the position of nodes to deliver a message to the destination. As a further advantage, position information enable the delivery of messages to all nodes in a given geographic region in a natural way. This type of service is called *geocasting* [NI97, Sto99].

Hence, message transmission in wireless ad-hoc networks can be preliminary classified in four groups:

Definition 1.

Unicast. *Unicast forwarding means one-to-one communication, i.e., one source transmits a message to a single destination.*

Multicast. *Multicast forwarding come into play when a node needs to send the same message to multiple destinations.*

Geocast. *Geocast forwarding is a special case of multicast that is used to deliver messages to a group of nodes situated inside a specified geographical area. Nodes may join or leave a multicast group as desired, but nodes can join or leave a geocast group only by entering or leaving the corresponding geographical region.*

Broadcast. *Broadcast forwarding sends a message to all nodes in the network. A node forwards a message to all of its neighbor nodes. Each of those neighbors in turn rebroadcasts the packet exactly one time and this continues until all reachable network nodes have received the message. This naive flooding protocol may cause a broadcast storm problem due to redundant rebroadcasts. Different schemes have been proposed to alleviate this problem by reducing redundant broadcasting [SW04].*

The biggest advantage of position information is the possibility to implement communication in a wireless ad-hoc network with totally localized approaches. Geographic routing algorithms (e.g., [Fin87, KK00]) only need the positions of the neighbors and the position of the destination node for forwarding. Hence, the memory requirements of such an algorithm are minimal. Geographic routing requires neither flooding nor the distribution of status information to further nodes than the single hop neighbor nodes. Routing with these algorithms is therefore more energy efficient and requires less bandwidth than traditional routing algorithms without the use of position information. Furthermore, as will be shown in Chapter 3, position information allows the localized computation of an efficient topology. Each node computes locally its contribution to the overall topology using only information about its single hop neighbor nodes. The communication in wireless ad-hoc networks hence becomes independent of the number of nodes in the network by the usage of position information. Actually, position based routing is only limited by the degree of node mobility.

2.5 Communication Graph

The communication graph on a set $\mathcal{S}$ of nodes is generated by having every node $s \in \mathcal{S}$ use its maximal transmission power. We assume that all nodes in the network have negligible difference in altitude, so they can be considered roughly in a plane.

Definition 2 (Communication Graph). *A wireless ad-hoc network can be modeled as a communication graph $CG(\mathcal{S}, \mathcal{E})$ in the plane, with a set of sites $\mathcal{S}$ and a set of edges $\mathcal{E}$. Each site s_i of the set $\mathcal{S} := \{s_0, \ldots, s_{N-1}\}$ represents a node of our wireless ad-hoc network. The total number of sites is $N = |\mathcal{S}|$.*

Definition 3 (Communication Edge). *A communication edge $e = (s_i, s_j)$, $e \in \mathcal{E}$, represents a wireless link of the communication graph $CG(\mathcal{S}, \mathcal{E})$. An edge (s_i, s_j) is present in $CG(\mathcal{S}, \mathcal{E})$ if and only if $\|s_i, s_j\|$ is less than or equals the maximal communication ranges of node s_i, where $\|s_i, s_j\|$ denotes the Euclidean distance between s_i and s_j.*

Definition 4 (Bidirectional Communiation Edge). *A communication edge (s_i, s_j) is called bidirectional if $\|s_i, s_j\|$ is less than or equals the maximal communication ranges of node s_i and the maximal communication ranges of node s_j. Otherwise, the communication edge is called unidirectional.*

Definition 5 (Neighborhood). *The set of nodes with which a node s, $s \in \mathcal{S}$ could directly communicate is called the neighborhood of a node, denoted by $\mathcal{N}(s)$. All nodes in $\mathcal{N}(s)$ are within the maximal communication range of node s. We assume n to be an upper bound on the nodes in the neighborhood $n \geq |\mathcal{N}(s)|$, $\forall s \in \mathcal{S}$.*

Definition 6 (Communication Range). *A wireless ad-hoc network is called* **homogeneous***, if all nodes use the same maximal communication range, and* **heterogeneous***, if the nodes use different maximal communication ranges.*

Definition 7 (Path). *A path $\mathcal{P}(s, t)$ from node s to node t, $s, t \in \mathcal{S}$, is an ordered set $(s = s_0, s_1, s_2, \ldots, s_k = t)$ of nodes such that there is an edge between consecutive nodes: $e = (s_i, s_{i+1})$ for $i = 0, \ldots, k-1$ with $e \in \mathcal{E}$.*

A deeper investigation of the properties required for topology control and routing in wireless ad-hoc networks is presented in Chapter 11.

2.6 Capacity

Capacity is a key characteristic of wireless ad-hoc networks. It represents the achievable data transmission rate that a network can support. Early work on multi-hop routing estimated that, because of limited transmission range, high spatial reuse of the spectrum would become possible. However, the seminal paper by Gupta and Kumar [GK00] showed that in a wireless ad-hoc network of N nodes, where each node is communicating with another node, the throughput per node is $\Theta(\frac{1}{\sqrt{N \log N}})$ assuming random node placement and communication pattern and $\Theta(\frac{1}{\sqrt{N}})$ assuming optimal node placement and communication pattern.

A multi-hop routing protocol requires that nodes cooperate to forward the messages through the network. The capacity of ad-hoc wireless networks is constrained by the interference between concurrent transmissions from neighboring nodes. Every node in the network acts simultaneously as a source, a destination

for some other node, as well as relays for others messages. The limited bandwidth available to each node must be divided for transmission of own messages and forwarding of messages from other nodes. The number of messages from other nodes increases with the number of nodes in the network. This effect could seriously limit the performance of multi-hop routing.

The following simplification of the analysis by Gupta and Kumar [GK00] estimates the per node capacity to be expected in an ad-hoc network. Nodes that are sufficiently distant can transmit concurrently; the total amount of data that can be simultaneously transmitted for one hop increases linearly with the total area of the ad hoc network. If the node density is constant, this means that the total one-hop capacity is $O(N)$. However, as the network grows larger, the number of hops between each source and destination may also grow larger, depending on communication patterns. The average path length grows with the spatial diameter of the network, or equivalently the square root of the area $O(\sqrt{N})$. With this assumption, the total end-to-end capacity is roughly $O(\frac{N}{\sqrt{N}})$, and the end-to-end throughput available to each node is $O(\frac{1}{\sqrt{N}})$. Hence, the throughput available to each node approaches zero as the number of nodes increases.

While the above discussion suggests that ad-hoc networks are fundamentally non-scalable, it may not reflect reality. Li et al. [LBC+01] have extended the work of Gupta and Kumar [GK00] by considering the impact of different traffic patterns on the scalability of per node throughput. Gupta and Kumar assume a random communication pattern: Each pair of nodes is equally likely to communicate, so that packet path lengths grow along with the physical diameter of the network. This assumption is probably reasonable for small networks. However, users in large networks may communicate mostly with physically nearby nodes: their neighbors in the same lecture hall of a university, or on the same floor of a building, or in the same company in a city. If local communication predominates, path lengths could remain nearly constant as the network grows, leading to constant per node available throughput.

Capacity is a challenge for every wireless ad-hoc network and each scientist, developer, and operator must find an answer to this challenge. However, the aim of our work is to investigate the fundamental limits of position-based communication in wireless ad-hoc networks and not the behavior under heavy network load. We hence assume a single point-to-point communication and analyze whether localized position-based routing with planar topologies can guarantee communication in arbitrary large networks.

Beklage nicht, was nicht zu ändern ist, aber ändere, was zu beklagen ist.

William Shakespeare (1564–1616)

Chapter 3

Topology Control

The topology plays a key role in the performance of routing algorithms. The technique to compute an appropriate topology is called *topology control* and forms an important part of research in wireless ad-hoc networking. Topology control algorithms maintain network connectivity while reducing energy consumption and improving network capacity. Topology control has become an active field of research in recent years. A good overview is given in the paper of Paolo Santi [San05].

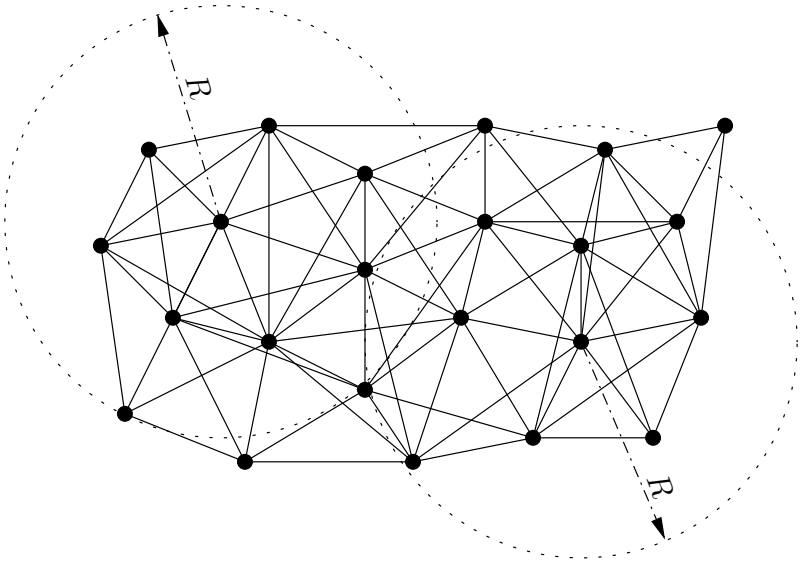


Figure 3.1: *Communication graph with maximum communication range R.*

The aim of the topology control algorithm is the computation of a subgraph of the communication graph $CG(\mathcal{S}, \mathcal{E})$. An example of the communication graph

can be seen in Figure 3.1. The topology returned by the topology control algorithm is defined as follows:

Definition 8 (Topology).
$T(S, E) \subseteq CG(\mathcal{S}, \mathcal{E}), \quad S = \mathcal{S},\ E \subseteq \mathcal{E}$

A topology control algorithm has two different tools for the computation of the topology $T(S, E)$:

- The connectivity in the topology can be changed by ignoring some possible communication links. The nodes decide whether an edge of the communication graph become parts of the topology or not. The decision can be done localized or centralized.
- The transmission power is the second tool to control connectivity. The communication range of a node decreases if the node reduce its transmission power. A smaller communication range reduces the number of nodes to which a node can directly communicate. When the power of the transmission is increased, a node can directly communicate with a larger number of neighbor nodes, but increased transmission power also increases the noise level in the network and thus the interference experienced by other nodes. It is possible that all nodes use the same transmission power, or each node looks on its own for the optimum transmission power.

The quality of a topology $T(S, E)$ can be evaluated according to several criteria including connectivity, energy-efficiency, transmission-efficiency, node degree, robustness against mobility, and planarity [Raj02].

3.1 Quality Properties of Topologies

Connectivity. The most basic requirement of a topology is that it be connected. More precisely, if there exist a path between node s and node t in $CG(\mathcal{S}, \mathcal{E})$, there must be a path between s and t in $T(S, E)$, $s, t \in S$. This criterion is essential to avoid partitions in the topology and to guarantee the delivery of messages by the routing algorithm. We can distinguish three different kinds of connectivity in a topology. We start with the weakest kind and move toward the strongest:

uni-connected. An uni-connected topology contains unidirectional links (Definition 4). It is not possible to guarantee the existence of a path from any node to any other node. A uni-connected topology does not necessarily achieve the connectivity specification presented above. A node can be a data sink only.

strongly connected. A topology is said to be strongly connected if there exists a path from any node to any other node in the network.

bi-connected. A topology is said to be bi-connected if all edges in the topology are bidirectional, i.e., if node s can directly communicate with node t then node t can also directly communicate with node s (Definition 4). Obvious, a bi-connected topology is also strongly connected.

Energy-efficiency. Since the topology $T(S, E)$ forms the underlying network for routing protocols, it is also desirable that energy-efficient paths between potential source-destination pairs exist. In the most common power-attenuation model, the power needed to support a communication link (s, t) is $\|s, t\|^{\alpha}$, where $\|s, t\|$ is the Euclidean distance between node s and node t, and α is a constant between 2 and 4 dependent on the wireless transmission environment. A sequence of short communication links requires less power than one long communication link, but the sequence of short links must approximate the long communication link. Therefore, in an energy-efficient topology $T(S, E)$ the length of the shortest path from node s to node t is only a constant factor larger than the length of the shortest path between these two nodes in $CG(\mathcal{S}, \mathcal{E})$. It can be shown that in a topology where the ratio between the shortest paths is constant, the ratio between the minimum power paths is also constant. See Section 5.3 for further information.

Transmission-efficiency (throughput). The throughput of a wireless ad-hoc network depends on, among other factors, the topology and the level of interference inherent to the topology. The particular interference depends on the relative positions of the nodes and their transmission radii. The limitation of data exchange to the nearest neighbors only reduces the interference level in the network enormously, because exclusive communication in the local neighborhood has no influence to nodes further away. Hence, the throughput of the wireless ad-hoc network will be increased. It is furthermore better to avoid the communication with far away nodes and to concentrate primary on near nodes, because the error-proneness of long links is many times higher than the error-proneness of short links [SZHK04]. Link-faults may cause retransmissions which require additional bandwidth and additional power, and the throughput of the wireless ad-hoc network decreases.

Node degree. The node degree in a topology should be small, ideally should the node degree be bounded from above by a constant. A small node degree can reduce the interference in the neighborhood and low interference improves the overall network throughout [KS78]. However, the node degree

should not be too small, because this produces communication bottlenecks and decreases the throughput of the network. Furthermore, a sparse network does not automatically reduce the interference. The edges must be carefully eliminated and sometimes is the topology with the shortest communication links not the topology with the lowest interference [BvRWZ04].

Robustness against mobility. In dynamic environments is it particularly important that topology control algorithms use only information about their single hop neighborhood to be robust with respect to node mobility. Information of double or more hop neighbors must be either broadcast in the network or each node has to maintain routing paths to all double or multi hop neighbors. Both methods are error-prone and waste a lot of bandwidth. Algorithms which require only information about single hop neighbors need no additional energy to forward received messages to other neighbors and the time-delay to distribute the information in the network is minimized. An efficient topology control algorithm needs an appropriate local approach to react on topology changes through node motion, appearance of new nodes and disappearance of erroneous nodes. A topology control algorithm is called *localized* if every node s computes locally its contribution to the overall topology using only information about its single hop neighbor nodes. A localized topology control algorithm is beneficial to make the topology fault-tolerant. Scalability is another benefit of localized algorithms. The restriction of information exchange with single hop neighbor nodes only, makes the algorithm independent from the total number of nodes in the network.

Planarity. Several routing protocols need a planar graph to guarantee the delivery of messages. Examples are Greedy Face Routing (GFG) [BMSU01], Greedy Perimeter Stateless Routing (GPSR) [KK00], Adaptive Face Routing (AFR) [KWZ02], and Greedy Other Adaptive Face Routing (GOAFR) [KWZ03b].

3.2 Topology Control Approaches

Nowadays, the term topology control is used for a multitude of methods to make the communication in wireless ad-hoc networks faster, securer and more efficient. In general, we can divide the different methods into three major paradigms — power-control approaches, hierarchical approaches and link-based approaches.

3.2.1 Power-control Topology Control

The primary goal of power-control approaches is to minimize the energy consumption in the wireless ad-hoc network. These approaches adjust the power on a per-node basis, so that sufficient connectivity is achieved and the power consumption is minimized (energy-efficiency). Topologies derived from power-control schemes often result in unidirectional links as a result of the different transmission ranges of single hop neighbors. Additional work is required to make the topology bi-connected. Figure 3.2 shows the topology of the communication graph shown in Figure 3.1 computed by a power-control topology control algorithm.

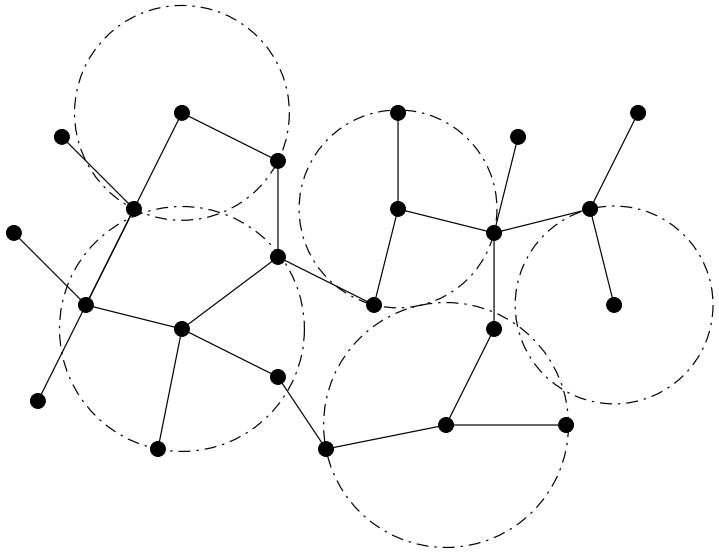


Figure 3.2: *Topology computed by a power-control topology control algorithm.*

In the work of Ramanathan and Rosales-Hain [RRH00], two centralized optimal algorithms were proposed for creating a bi-connected static network with the objective of minimizing the maximum transmitting power for each node. Additionally, two distributed heuristics, LINT (local information no topology) and LILT (local information link-state topology), were proposed for adaptively adjusting node transmitting power to maintain a connected topology in response to topological changes. But, neither LINT nor LILT can guarantee the connectivity of the network. Ramanathan and Rosales-Hain showed that a topology which minimizes the maximum transmitter power allocated to any node can be constructed in polynomial time.

Das and Mesbahi [DM05] consider a different version of the power efficient topology optimization problem, that of minimizing the total transmit power, as opposed to minimizing the maximum transmitter power. Minimizing the total transmit power has also the effect of limiting the total interference in the network. The problem of finding the solution which requires minimum power is called the *minimum range assignment problem*. This problem consists of assigning transmission ranges to the nodes of a wireless ad-hoc network so as to minimize the total power consumption provided that the transmission ranges assigned to the nodes ensure the connectivity of the network. The computation of a topology with minimum power effort, either with unidirectional or with bidirectional links, is NP-hard [CPS99, LLM+02]. However, a topology with minimum power effort is not necessarily the practical optimum, because such a topology can be very sparse. Adequate connectivity in the topology is an important property to improve the throughput in a wireless ad-hoc network.

3.2.2 Hierarchical Topology Control

Hierarchical topology control selects a subset of the network nodes to serve as the network backbone over which essential network functions are supported (e.g., routing [KVCP97]). This approach to topology control is often called clustering, and consists of selecting a set of clusterheads in a way such that every node is associated with a clusterhead, and clusterheads are connected with one another directly or by means of gateways, so that the union of clusterheads and gateways constitute a connected backbone [BGLA03]. See Figure 3.3 for an example of a topology computed by an hierarchical topology control algorithm. Once elected, the clusterheads and the gateways help to reduce the costs of maintaining topology and can simplify routing. The election of clusterheads is the same problem as finding a minimum connected dominating set in graph theory. A dominating set is a subset of nodes such that each node is either in the subset or adjacent to at least one node in the subset. The problem of computing a minimum dominating set is known to be NP-hard even when the complete network topology is available. In wireless ad-hoc networks, the difficulty of acquiring the complete network topology makes it impossible to compute the minimum dominating set. Instead, a minimal dominating set is usually pursued based on various heuristics. Different heuristics have been used to from clusters and to elect clusterheads. Some approaches utilized the node identifiers to elect the clusterhead (e.g., Gao et al. [GGH+01a]). The node degree is another commonly used heuristic in which nodes with higher degrees are more likely to become clusterheads (e.g., Jia et al. [JRS02]). In the work of Bao and Garcia-Luna-Aceves [BGLA03] the battery life and the mobility of nodes are combined to a value to decide if the node becomes a clusterhead or not.

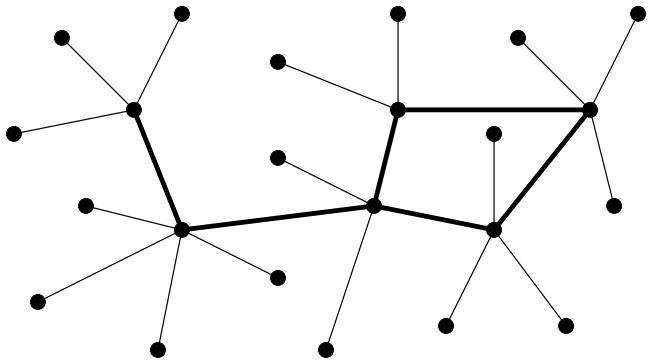


Figure 3.3: *Topology computed by a hierarchical topology control algorithm.*

The usage of connected clusterheads to support high-level network functions yields increased traffic on the backbone. The interference near the backbone is significantly higher than the average interference in the network. The throughput in the network decreases. A hierarchical topology control approach creates a topology similar to the topology of a cellular network. The topology looses the benefits of ad-hoc networks, e.g., spreading of network traffic, but does not acquire the benefits of cellular networks, because the elected clusterheads are not necessarily efficient and powerful nodes. Furthermore, the election of clusterheads reduces the fault-tolerance in the network (single-point of failure) and requires a lot of additional computation to make the approach applicable in dynamic environments.

3.2.3 Link-based Topology Control

Link-based topology control selects a subset the communication links to form an efficient topology. In the communication graph $CG(\mathcal{S}, \mathcal{E})$ each network node has a lot of communication links to other network nodes, the network is perhaps fully connected, i.e., each node can potentially communicate with all other nodes in the network. Hence, the task of link-based topology control is a reasonable restriction of the available communication links to a small number of beneficial

links. An important parameter for the restriction of the communication links is energy efficiency. A sequence of short communication links requires less power than one long communication link. Short communication links are more reliable. The susceptibility to communication faults increases with increasing distance between the neighbor nodes. The limitation of data exchange to the nearest neighbors only reduces also the interference level in the network enormously, because exclusive communication in the local neighborhood has no influence to nodes further away. Hence, the throughput of the network will be increased. Furthermore is it easier to achieve bidirectional communication links with this approach than with the power-control topology control approach. However, the communication links must be carefully eliminated, because connectivity is still the most important requirement for a topology.

The topology control approach presented by Rodoplu and Meng [RM99] selects those neighbors for which direct transmission requires less power than if an intermediate node is used to retransmit the message. They introduced for their approach the notion of relay region and enclosure. For any node s_i that intends to transmit to node s_j, node s_j is said to lie in the relay region of a third node s_k, if node s_i will consume less power when it chooses to relay through node s_k instead of transmitting directly to node s_j. Each node uses the position of the neighbor nodes and a simple radio propagation model to compute the relay regions. The enclosure of node s_i is then defined as the union of the complement of relay regions of all the neighbor nodes of node s_i. In the resulting topology each node has only communication edges to the nodes within its enclosure and the edges are directed. The local enclosure graphs constructed for individual nodes form globally a strongly connected graph guaranteed to contain the minimum-energy paths for all pairs of nodes. A position service is required which provides the nodes with position information.

The approach presented in [BvRWZ04] uses as decision criterion the interference of the communication links. The minimization of the interference in a wireless ad-hoc network lowers node energy consumption by reducing the number of collisions and consequently packet retransmission. Burkhart et al. propose three algorithms for the computation of interference-optimal topologies, e.g., the *Low Interference Forest Establisher* (LIFE) algorithm is a centralized algorithm that computes a topology with least possible interference.

A general link-based topology control protocol is the *XTC topology control algorithm* [WZ04]. In the XTC protocol each node computes a total order over all its neighbors in the communication graph. This order is intended to reflect the quality of the edges to the neighbor nodes. The quality of an edge is defined

by a given cost metric. Popular cost metrics are transmission power, Euclidean distance, signal attenuation, packet arrival rate, interference etc. . However, the cost metric must by symmetric, i.e., the weight of an edge from s_i to s_j must be the same as from s_j to s_i. During the computation step, a node s_i ignores an edge to s_j if there exists a nodes s_k such that for s_i and s_j an edge to s_k is better. It is shown in [WZ04] that the resulting topology is bi-connected provided $CG(\mathcal{S}, \mathcal{E})$ is.

The *Cone-Based Topology Control* (CBTC) approach, proposed by Wattenhofer et al. [WLBW01], takes a parameter α, and each node s, $s \in \mathcal{S}$, determines a power level $p_{s,\alpha}$ such that in every cone of α degrees surrounding s there is at least one node reachable with $p_{s,\alpha}$. Each node starts with a small initial power and gradually increases it until the above condition is satisfied. The topology $T(S, E)$ contains all edges for each node s that was found in the search process. The authors also prove that, if $\alpha \leq \frac{2\pi}{3}$ then the resultant topology is strongly connected provided the communication graph $CG(\mathcal{S}, \mathcal{E})$ is strongly connected. Later, the critical cone angel that guarantees strong connectivity was found to be $\frac{5\pi}{6}$ by Li et al. [LHB$^+$01]. One issue with the algorithm is the determination of the suitable initial power level and the increment of power level at each step. The choice of these two parameters may have significant impact on the number of overhead messages needed to create the desired topology. The cone-based topology control approach [WLBW01] requires directional information of incoming signals from neighbor nodes. Therefore, the nodes have to use multiple antennas to provide *Angel of Arrival (AoA)* information. Similar approaches use position information instead of directional information.

An interesting group of link-based topology control approaches use the position of nodes and the construction rules of proximity graphs for the computation of the topology. A proximity graph is a graph derived from the geometry of a set of points. Well known examples of proximity graphs are Relative neighborhood graph, Gabriel graph, and Delaunay triangulation. They all use the concept of adjacency relations between points. A detailed description of different proximity graphs is given in Chapter 4.

A topology based on a proximity graph inherits the properties of the proximity graph. These properties are connectivity, the spanning factor and planarity. Remember, planarity is an important property for some routing approaches. It is still impossible to compute a planar topology without the position of the nodes. Figure 3.4 shows a topology computed with the construction rule of the Gabriel graph.

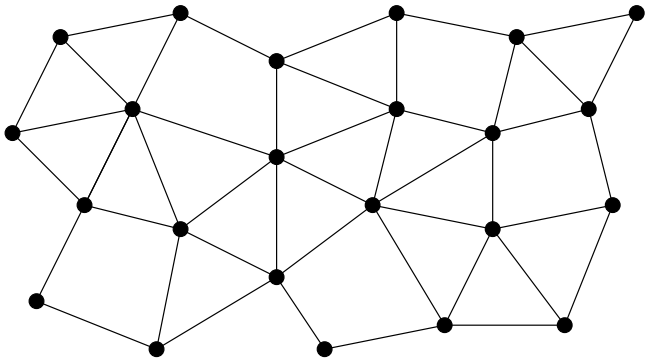


Figure 3.4: *Topology computed by a link-based topology control algorithm.*

Link-based topology control algorithms that use the construction rules of proximity graphs for the computation of the topology are in the following referred as position-based topology control algorithms. Particular publications on position-based topology control algorithms are presented in Section 5.6.

Das Unverständlichste am Universum ist im Grunde, das wir es verstehen.

Albert Einstein (1879–1955)

Chapter 4

Proximity Graphs

Wireless ad-hoc networks use proximity graphs as the basic "construction plan" for the computation of the topology. We denote topology control algorithms that use proximity graphs for the computation as position-based topology control algorithms (Chapter 3). Computation algorithms and the features of proximity graphs are well studied in literature. This chapter is a short introduction to computational geometry to make the reader familiar with the notations and the properties of proximity graphs.

In the following we consider a finite set $S := \{s_0, \ldots, s_{n-1}\}$ of points in the plane whose elements are referred to as sites, in order to distinguish them from ordinary points $\{p, q, \ldots\}$ in the plane.

4.1 Proximity Graphs

Proximity graphs represent neighbor relationships between geometric points in the Euclidean plane. Intuitively speaking, a proximity graph on a finite set S of sites in the Euclidean plane, $S \subset \mathbb{R}^2$, is obtained by connection pairs of sites of S with line segments if the sites are considered to be close in some sense. Different definitions of closeness give rise to different proximity graphs. The most famous examples of proximity graphs are the *Euclidean minimum spanning tree* ($EMST(S)$), the *Relative neighborhood graph* ($RNG(S)$) [Tou80], the *Gabriel graph* ($GG(S)$) [GS69], the *Yao graph* ($YG_k(S)$) [Yao82] and the *Delaunay triangulation* ($DT(S)$) [Del32, Del34].

Definition 9 (Euclidean minimum spanning tree). *A Euclidean minimum spanning tree ($EMST(S)$) of a set S of sites in the plane is a tree of minimum total edge length connecting all sites $s \in S$.*

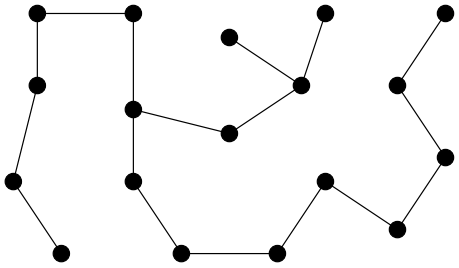


Figure 4.1: *Euclidean minimum spanning tree*

The edges are bidirectional (Definition 4 and it is impossible to remove an edge from the graph without loosing connectivity. The $EMST(S)$ is loop free, adding only one edge to $EMST(S)$ creates a loop. An example of an Euclidean minimum spanning tree connecting a given set of sites is shown in Figure 4.1. The Euclidean minimum spanning tree on a given set of sites is not necessarily unique, because two or more spanning trees can have the same minimum total edge length.

Definition 10 (Relative neighborhood graph). *An edge (s_i, s_j) is in $RNG(S)$ [Tou80] if and only if the lune of s_i and s_j does not contain any site of S. The lune of a pair of sites $s_i, s_j \in S$ is the intersection of two open discs of radius $\|s_i, s_j\|$, one centered at s_i and the other centered at s_j.*

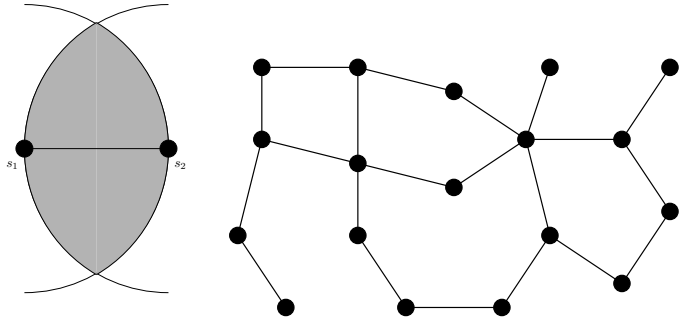
The Relative neighborhood graph is unique on a given set of sites and the edges are bidirectional. An example of the Relative neighborhood graph $RNG(S)$ on a set S is shown in Figure 4.2.

Definition 11 (Gabriel graph). *The Gabriel graph [GS69] $GG(S)$ of a set S consists of all edges (s_i, s_j) with the property that no other site of S is inside or on the circle through s_i and s_j which has $\|s_i, s_j\|$ as a diameter.*

Just as the Relative neighborhood graph the Gabriel graph is unique on a given set of sites and the edges are bidirectional. Figure 4.3 shows the construction rule of the Gabriel graph and an example on a set of sites.

The Yao graph is the geometrical description of the *Cone-Based Topology Control* (CBTC) algorithm proposed by Wattenhofer et al. [WLBW01].

Definition 12 (Yao graph). *Let S be a set of sites in the Euclidean plane and let $k \in \mathbb{N}$. The directed Yao graph of S [Yao82] of parameter k is denoted $\overrightarrow{YG}_k(S)$, and is defined as follows. At each site $s_i \in S$, any k equally separated rays originated at s_i define k equal cones with angle $\Theta = 2\pi/k$. In each cone, choose the closest site s_j to s_i, if there is any, and add the corresponding directed*



(a) *Critical section of edge* (s_1, s_2) *in* $RNG(S)$

(b) *Relative neighborhood graph*

Figure 4.2: *Relative neighborhood graph*

edge (s_i, s_j) *to* $\overrightarrow{YG}_k(S)$*. Let* $YG_k(S)$ *be the corresponding bidirectional graph, resulting from ignoring the direction of each edge in* $\overrightarrow{YG}_k(S)$*. The Yao graph is guaranteed to be connected if* $k \geq 6$ *(Corollary 1).*

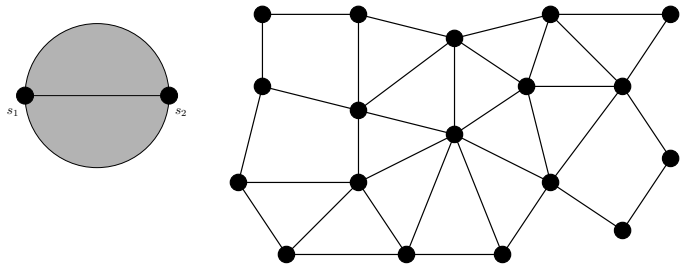
The Yao graph is not unique, because a site s_i can have two sites in a cone s_j and s_k, $s_i, s_j, s_k \in S$, with the same distance to s_i. Such ties are broken arbitrarily. An example of the Yao graph with $k = 6$ can be found in Figure 4.4. Some researchers used a similar construction named the Θ-Graph [KG92, Luk99], the difference is that it chooses in each cone the edge which has the shortest projection on the halfline which bisects the cone instead of the shortest edge in the cone.

Definition 13 (Delaunay triangulation). *A triangulation of a set* S *of sites in the plane is called a Delaunay triangulation* $DT(S)$ *of* S *[Del32, Del34], if the circumcircle of each of its triangles does not contain any site of* S *in its interior*[1].

The Delaunay triangulation is often refereed to as Delaunay diagram. Definition 13 does not consider the case where all sites are colinear. The following Definition 14 is a little bit stronger and more convenient for our purpose:

Definition 14 ("empty circle" property). *An edge* (s_i, s_j) *is a Delaunay edge if there exists a circle through* s_i *and* s_j *so that no site of* S *lies properly inside this circle.*

[1]We consider that the interior of a circle is an open disc, i.e., the boundary is excluded.



(a) *Critical section of edge* (s_1, s_2) *in* $GG(S)$

(b) *Gabriel graph*

Figure 4.3: *Gabriel graph*

Figure 4.5 shows $DT(S)$ on a given set S of sites. The Delaunay triangulation is not unique if four or more sites are cocircular, ties are broken arbitrarily. The edges in the Delaunay triangulation are bidirectional.

4.2 Properties of Proximity Graphs

4.2.1 Relations between Proximity Graphs

Proximity graphs are related to each other. The following relationships among the different proximity graphs hold for any finite set S of sites in the plane.

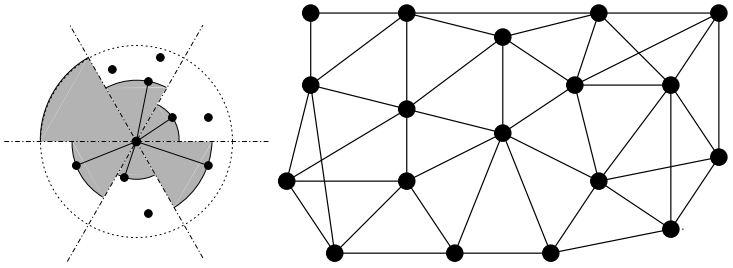
Lemma 1. *The relationships between Euclidean minimum spanning tree, Relative neighborhood graph, Gabriel graph, Delaunay triangulation and Yao graph are:*

$$EMST(S) \subseteq RNG(S) \subseteq GG(S) \subseteq DT(S) \tag{4.1}$$

$$EMST(S) \subseteq RNG(S) \subseteq YG_k \quad \textit{for any} \quad k \geq 6 \tag{4.2}$$

Proof.

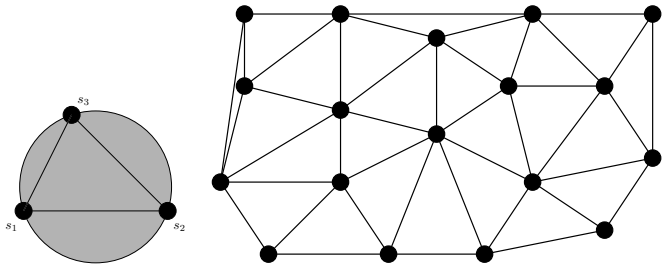
$GG(S) \subseteq DT(S)$**.** Edge (s_i, s_j) is in the Gabriel graph if there are no other sites than s_i and s_j in the closed disc with diameter $\|s_i, s_j\|$. Obviously, the ”empty circle” property (Definition 14) is also fulfilled if the closed disc with diameter $\|s_i, s_j\|$ is empty. Therefore all edges in the Gabriel graph are in the Delaunay triangulation.



(a) *Critical sections of a site in* $YG_6(S)$

(b) *Yao graph*

Figure 4.4: *Yao graph*



(a) *Critical section of triangle* (s_1, s_2, s_3) *in* $DT(S)$

(b) *Delaunay triangulation*

Figure 4.5: *Delaunay triangulation*

$RNG(S) \subseteq GG(S)$**.** For the sake of contradiction, we assume that there is an edge (s_i, s_j) in $RNG(S)$ that is not in $GG(S)$. By definition of $GG(S)$ is $(s_i, s_j) \notin GG(S)$ only if there is a site s_k in the closed disc with diameter $\|s_i, s_j\|$. $\|s_i, s_k\| < \|s_i, s_j\|$ and $\|s_j, s_k\| < \|s_i, s_j\|$. This is a contradiction to the existence of $(s_i, s_j) \in RNG(S)$

$EMST(S) \subseteq RNG(S)$**.** One way to construct the Euclidean minimum spanning tree is to add edges in the order of their length to $EMST(S)$ if it does not create a cycle with previously added edges. For the sake of contradiction, we assume that there is an edge (s_i, s_j) in $EMST(S)$ that is not in $RNG(S)$. By definition of $RNG(S)$ is $(s_i, s_j) \notin RNG(S)$ only if there is a site s_k in the interior of the lune of s_i and s_j. Obviously, edges

(s_i, s_k) and (s_j, s_k) are shorter than edge (s_i, s_j). No matter whether the edge (s_i, s_k) is in $RNG(S)$, we know that there is a path in $RNG(S)$ (could be edge (s_i, s_k)) connecting s_i and s_k using edges with length at most (s_i, s_k). Similarly, the same property holds for edge (s_j, s_k). Thus, when we add edge (s_i, s_j) to $EMST(S)$, it will create a cycle with edges already in $EMST(S)$ since the edges in path from s_i to s_j (via site s_k) definitely are shorter than edge (s_i, s_j), i.e., added before edge (s_i, s_j). This is a contradiction to the existence of (s_i, s_j) in $EMST(S)$.

$RNG(S) \subseteq YG(S)_k$ **for any $k \geq 6$.** Prove by contradiction, assume that there is an edge (s_i, s_j) in $RNG(S)$ that is not in $YG_6(S)$. By definition of $YG_k(S)$ is $(s_i, s_j) \notin YG_k(S)$ only if there is a site s_k in $cone(s_i, s_j)$ and $\|s_i, s_k\| \leq \|s_i, s_j\|$. In $YG_6(S)$ is $cone(s_i, s_j)$ the sector bounded by two rays originated at s_i with angle $\Theta = \pi/3$ and containing site s_j. By the *Law of cosines* is

$$\|s_j, s_k\|^2 \leq \|s_i, s_j\|^2 + \|s_i, s_k\|^2 - 2\|s_i, s_j\|\|s_i, s_k\| \cos \tfrac{\pi}{3}$$

$$\|s_j, s_k\|^2 \leq \|s_i, s_j\|^2 + \|s_i, s_k\|^2 - \|s_i, s_j\|\|s_i, s_k\|$$

$$\|s_j, s_k\|^2 \leq \|s_i, s_j\|^2 + \|s_i, s_k\|^2 - \|s_i, s_k\|\|s_i, s_k\|$$

$$\|s_j, s_k\| \leq \|s_i, s_j\|. \tag{4.3}$$

From (4.3) follows that site s_k is in the interior of the lune of s_i and s_j. This is a contradiction to the existence of (s_i, s_j) in $RNG(S)$.

□

Corollary 1. *$YG_k(S)$ is connected if $k \geq 6$, because $EMST(S) \subseteq YG_k$ for any $k \geq 6$.*

4.2.2 Planarity

We show in the following lemma the planarity of the Euclidean minimum spanning tree, the Relative neighborhood graph, the Gabriel graph and the Delaunay triangulation. The Yao graph is not guaranteed to be planar [Yao82].

Lemma 2. *$EMST(S)$, $RNG(S)$, $GG(S)$ and $DT(S)$ are planar graphs.*

Proof. We show that no two edges may cross in the Delaunay triangulation. Suppose two crossing edges, (s_1, s_2) and (s_3, s_4), appear in the Delaunay triangulation. By Definition 14, there is a circle through s_1 and s_2 not containing s_3 or s_4, and likewise there is a circle through s_3 and s_4 not containing s_1 or s_2. If these properties are fulfilled, the two distinct "circles" must intersect at four points which is impossible. If four or more sites are cocircular, the circles intersect at an infinite number of points, but each algorithm for the computation of the Delaunay triangulation has a method to break such ties and to establish a correct triangulation. Due to Lemma 1, the Euclidean minimum spanning tree, the Relative neighborhood graph and the Gabriel graph are also planar. □

Each planar triangulation (e.g., the Delaunay triangulation) is not only denser than the Relative neighborhood graph and the Gabriel graph, but also denser than any other planar subgraph.

Lemma 3. *(Edelsbrunner [Ede87]) A triangulation of a set S of sites is a maximum planar subdivision: no edge connecting two sites can be added to the triangulation without destroying planarity.*

4.2.3 Spanning Property

Perhaps the most basic requirement of a proximity graph is connectivity. More precisely, we require that any two sites of S are connected by a path. The *length stretch factor* defines the maximum ratio of the minimum path length between two sites over the Euclidean distance. Consider a graph G over a finite non empty set $S \in \mathbb{R}^2$. For each pair of sites s_i, s_j, the length of the shortest path connecting s_i and s_j measured by Euclidean distance is denoted by $\Pi_G(s_i, s_j)$, while the direct Euclidean distance is $\|s_i, s_j\|$.

Definition 15 (Length stretch factor). *The length stretch factor of the graph G is defined as the maximum ratio of the shortest path length connecting any pair of sites in G to their distance.*

$$\sigma_G := \max_{(s_i,s_j)} \frac{\Pi_G(s_i, s_j)}{\|s_i, s_j\|} \qquad \forall s_i, s_j \in S \tag{4.4}$$

Some researchers denote the length stretch factor as *dilation ratio* or *spanning ratio*. The fully connected graph has $\sigma = 1$, but is less interesting because the graph is not planar and the number of edges is not linear but quadratic in N. If the graph is not connected, then σ is infinity, so it is reasonable to focus on connected graphs only.

The bounds on the length stretch factors in the following Lemmata 5–8 are upper bounds on all possible sets of sites in $\mathbb{R}^2$. It is impossible to find a set $S \in \mathbb{R}^2$ which has a higher spanning ratio than the presented bounds, although there may be sets where the spanning ratio is less.

Lemma 4. *The length stretch factor of a Euclidean minimum spanning tree is at most $\sigma EMST = N - 1$ (see [LWW01]).*

Corollary 2. *In any proximity graph that contains the Euclidean minimum spanning tree the length stretch factor is at most $N - 1$.*

Lemma 5. *The length stretch factor of a Relative neighborhood graph is at most $\sigma RNG = N - 1$ (see [WLMN+03]).*

Lemma 6. *The length stretch factor of a Gabriel graph is at most $\sigma_{GG} = \sqrt{N-1}$ (see [WLMN+03]).*

Lemma 7. *The length stretch factor of a Yao graph is at most $\sigma_{YG_k} = \frac{1}{1-2\sin\frac{\pi}{k}}$ (see [KG92]).*

Lemma 8. *The length stretch factor of the Delaunay triangulation has a constant upper bound, which is at least $\frac{\pi}{2}$ and at most $\frac{2\pi}{3\cos(\frac{\pi}{6})} \approx 2.42$ [KG92]. It is conjectured that the upper bound of the length stretch factor of the Delaunay triangulation is $\sigma_{DT} = \frac{\pi}{2}$.*

A graph with a length stretch factor bounded by a constant is called a *spanner*.

Definition 16 (Spanner). *A graph G is called a spanner, if $\Pi_G(s_i, s_j)$ is no more than a constant factor larger than $\|s_i, s_j\|$, for all $s_i, s_j \in S$.*

The Yao graph for $k > 6$ and the Delaunay triangulation are spanners, whereas the length stretch factors of the Euclidean minimum spanning tree, Relative neighborhood graph and Gabriel graph are linear in N.

4.2.4 Edge Power

It is both important and interesting to know lower and upper bounds for the number of edges in proximity graphs. If $N = |S|$ denotes the total number of sites, the maximum number of edges on a set of N sites is $\frac{N(N-1)}{2}$.

Definition 17 (Sparse Graph). *In a sparse graph the number of edges is linear.*

We show in the following that the number of edges in the presented proximity graphs are in $O(N)$.

Corollary 3. *The Euclidean minimum spanning tree $EMST(S)$ has exactly $N-1$ edges.*

Lemma 9. *The Delaunay triangulation of a set S of N sites in the plane, not all collinear, has at most $2N-5$ triangles and $3N-6$ edges.*

Proof. Let N be the number of sites, N_e the number of edges, and N_f the number of faces. From Euler's Theorem for connected planar graphs, we have

$$N - N_e + N_f = 2.$$

Let t denote the number of triangles in the triangulation. Each triangle is a face and there is also the single unbounded outer face. Hence, the number of faces $N_f = t + 1$. Let $k \geq 3$ denote the number of sites on the Convex hull of S. Each triangle has 3 edges and the single unbounded face has k edges. Since each edge belongs to exactly 2 faces, we have $3t + k = 2N_e$. Substituting the values of N_f and N_e, we obtain:

$$t = 2N - 2 - k \quad \text{and} \quad N_e = 3N - 3 - k.$$

□

Due to the results of Lemma 1 we have immediately that

$$|EMST(S)| \leq |RNG(S)| \leq |GG(S)| \leq |DT(S)|.$$

By the results of Corollary 3 and Lemma 9 we have that

$$N - 1 \leq |RNG(S)| \leq |GG(S)| \leq 3N - 6.$$

A more detailed analysis gives tighter bounds:

$$|RNG(S)| \leq 3N - 10 \text{ [Urq83]} \quad \text{and} \quad |GG(S)| \leq 3N - 8 \text{ [MS80].}$$

In the Yao graph the number of edges is bounded with

$$|YG_k(S)| \leq kN.$$

4.2.5 Site Degree

The Euclidean minimum spanning tree, the Relative neighborhood graph, the Gabriel graph, the Yao graph and the Delaunay triangulation are sparse (see Section 4.2.4). This implies that they have a constant average site degree. A tighter bound on the average site degree can be given for planar graphs.

Lemma 10. *The average site degree of*

$$EMST(S),\ RNG(S),\ GG(S),\ \textit{and}\ DT(S)$$

is at most 6.

Proof. Let N be the number of sites, N_e the number of edges and let D_i be the site degree of site s_i, $0 \leq i \leq N-1$. Lemma 2 shows that $EMST(S)$, $RNG(S)$, $GG(S)$ and $DT(S)$ are planar. As a result of Euler's formula (see Lemma 9) is the number of edges in a planar graph at most $3N-6$. Since each edge belongs exactly to 2 sites,

$$\sum_{i=0}^{N-1} D_i = 2N_e \leq 2(3N-6) = 6N-12.$$

Than the average site degree of a planar graph is given by,

$$D_{average} \leq \frac{6N-12}{N} = 6 - \frac{12}{N}.$$

□

Additionally, for the Euclidean minimum spanning tree is the average site degree equal to the maximum site degree.

Lemma 11. *The maximum site degree of* $EMST(S)$ *is* 6.

Proof. Assume a site s in the $EMST(S)$ with site degree larger than 6. At least one angle α between two subsequent edges, w.l.o.g. edge (s, s_1) and edge (s, s_2), is less than $\pi/6$. The edges (s, s_1) and (s, s_2) are part of $EMST(S)$ if $\|s, s_1\| \leq \|s_1, s_2\|$ and $\|s, s_2\| \leq \|s_1, s_2\|$. By the *law of sines*, the longest side of a triangle is opposite the largest angle. Since $\alpha < \pi/6$, α cannot be the largest angle in the triangle, and (s_1, s_2) cannot be longer than (s, s_1) and (s, s_2). Hence, $EMST(S)$ contains (s_1, s_2) instead of (s, s_1) or (s, s_2), and the site degree of s is at most 6. □

The maximum site degree of $RNG(S)$, $GG(S)$, $YG_k(S)$ and $DT(S)$ is not bounded by a constant. The maximum out-degree of a site in the Yao graph is k, but the maximum in degree could also be as large as $N-1$. Figure 4.6 shows the linearity of the maximum site degree of $RNG(S)$, $GG(S)$, $YG_k(S)$ and $DT(S)$. The instance consist of $N-1$ sites $s_i \in S$, $1 \leq i \leq N-1$, lying on the circle centered at site s, $s \in S$. Thus each edge (s, s_i) belongs to $RNG(S)$, $GG(S)$, $YG_k(S)$ and $DT(S)$. Unfortunately, it has been shown that no proximity graph with constant maximum site degree and constant bounded length stretch factor

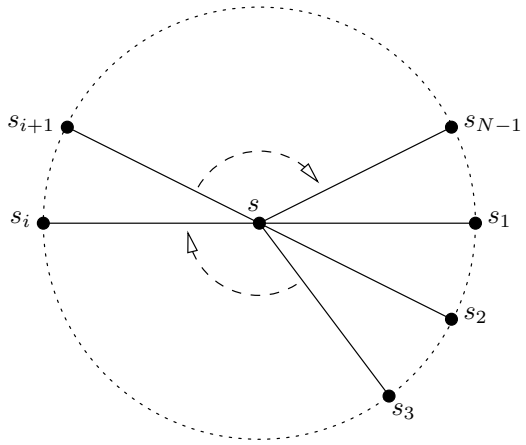


Figure 4.6: *Maximum site degree of* $RNG(S)$, $GG(S)$, $YG_k(S)$ *and* $DT(S)$

exists [WLF03].

The site degree is an important parameter for the connectivity of a graph. It can be shown that a graph with minimum site degree k is also k connected with high probability [Pen99]. A graph is said to be k-connected ($k = 1, 2, 3, \dots$) if for each site pair there exist at least k mutually independent paths connecting them. Equivalently, a graph is k-connected if and only if no set of $(k-1)$ sites exists whose removal would disconnect the graph. Similarly, a graph is called k-edge-connected if and only if there are at least k edge-disjoint paths between every pair of nodes. If a graph is k-connected, then it is also k-edge-connected, but the reverse implication is not necessarily true.

4.3 Computation of Proximity Graphs

The problem of computing the Euclidean minimum spanning tree, the Relative neighborhood graph, the Gabriel graph, the Yao graph, and the Delaunay triangulation of a set of sites S can be reduced to sorting. The lower bound of sorting is at least $\Omega(N \log N)$, therefore the complexity of computing a proximity graph is at least $\Omega(N \log N)$ [PS85].

Two algorithms for the computation of $DT(S)$ in two dimensions are the divide-and-conquer algorithm proposed by Guibas and Stolfi [GS85] and the plane-sweep approach presented by Fortune [For87]. Both algorithms achieve the optimal $O(N \log N)$ worst-case time complexity for the computation of the Delaunay triangulation of N sites. The problem of efficiently computing subgraphs of

the Delaunay triangulation of a set of sites is the subject of a lot of literature in computational geometry. A common method for the computation of $EMST(S)$, $RNG(S)$, and $GG(S)$ in two dimensions is by discarding edges from Delaunay triangulation. One of the first papers is due to Chariton and Tarjan [CT76] who show a $O(N)$-time algorithm to compute the Euclidean minimum spanning tree from the Delaunay triangulation with N sites. Supowit [Sup83] gave the first optimal algorithm for computing $RNG(S)$: it extracts $RNG(S)$ from $DT(S)$ in time $O(N \log N)$. The Gabriel graph can be computed from the Delaunay triangulation in $O(N)$-time by using the algorithm of Matula and Sokal [MS80]. Algorithms for the computation of the Yao graph are presented by Ruppert and Seidel [RS91] and by Arya et al. [AMS94]. Both approaches use plane sweeps and compute $YG_k(S)$ in $O(N \log N)$ time. Different algorithms for the computation of the Delaunay triangulation are given in Appendix A.

Ordnung marschiert mit gewichtigen und gemessenen Schritten, Unordung ist immer in Eile.

Napoleon Bonaparte (1769–1821)

Chapter 5

Position-based Topology Control

A wireless ad-hoc network needs a suitable topology to improve the performance and the efficiency of the routing algorithm. Position-based topology control algorithms combine two different topics of computer science – i.e., wireless ad-hoc networks and computational geometry. The aim of a position-based topology control approach is the computation of a topology as similar as possible to a proximity graph. In computational geometry, the proximity graphs Euclidean minimum spanning tree, Relative neighborhood graph, Gabriel graph, Yao graph, and Delaunay triangulation are well known. Chapter 4 gave an introduction to these proximity graphs and presents some important properties. Position-based topology control algorithms require a position service to provide the nodes in the wireless ad-hoc network with location information.

In the following we denote nodes with the letter s to stay uniform with the proximity graph definitions of Chapter 4 and to avoid confusions with the number of nodes. Remember: n is the maximum number of nodes in the neighborhood and N is the total number of nodes in the network.

5.1 Distributed Computation of Position-based Topologies in Sparse Networks

The concepts of Euclidean minimum spanning tree, Relative neighborhood graph, Gabriel graph, Yao graph, and Delaunay triangulation are originated in Computational Geometry, but these proximity graphs are not directly applicable to wireless ad-hoc networks. The input data of proximity graphs are only the geometric positions of a set of sites. The edges of the proximity graphs are determined by the computation algorithm and can be arbitrarily long, i.e., exceed the transmission ranges of the adjacent nodes.

Position-based topology control algorithms use the construction rules of proximity graphs for the computation of a topology for a wireless ad-hoc networks. A site represents a node and an edge represents a communication link in the wireless ad-hoc network. Communication links in wireless ad-hoc networks are confined by the maximum communication range, whereas the length of edges in proximity graphs are only depending on the geographic positions. Only a fully connected network offers the same qualities than the edge independent environment present in computational geometry. In a fully connected network it is possible for each node to communicate with all other nodes in the network directly. Each node knows the position of all other nodes and the computation of the topology yields a topology which is equal to the proximity graph. Unfortunately, a fully connected network is extremely unlikely for wireless ad-hoc networks.

Each edge in the communication graph $CG(\mathcal{S}, \mathcal{E})$ (Definition 2) of a wireless ad-hoc network represents a communication link. The communication graph is a subgraph of the fully connected network. Normally, a node s can only communicate with a small subset of the nodes. This subset, denoted by $\mathcal{N}(s)$, is called the neighborhood of s (Definition 5). We assume in the following an obstacle free communication environment and symmetric communication links (i.e., the sender and the receiver should observe the same channel properties such as interference, path loss, and fading). The communication ranges at all nodes are circular and homogeneous, i.e., all nodes have the same communication range. Hence, all nodes can communicate with all nodes within the communication range and the neighborhoods are consistent (i.e., $\forall s_i \in \mathcal{N}(s) : s \in \mathcal{N}(s_i)$). A generalized analysis of these requirements and the behavior of position-based topology control algorithms with heterogeneous communication ranges is given in Chapter 11.

Proximity graphs are computed centralized at some unit. Wireless ad-hoc networks are by definition completely decentralized, so a distributed computation of the topology at the nodes is required. Efficient and scalable algorithms for wireless networks may only use information about the local neighborhood. If a node needs information from double or multi hop neighbors, some kind of routing is required to deliver the information to the node. This wastes a lot of bandwidth and energy. Security and fault-tolerance are other important reasons to restrict the information exchange (for the computation of the topology) to the single hop neighborhood, because a malicious node can corrupt the information of a lot of double hop neighbors otherwise. Hence, the computation of the topology should be done localized, only with information of the single hop neighborhood. Localized computation guarantees scalability, because the algorithm is independent of the total number of nodes in the wireless ad-hoc network.

In a distributed position-based topology control algorithm each node $s_i \in S$ computes a *local topology* only with the nodes of its neighborhood $\mathcal{N}(s_i)$, denoted by $T_{PG}(\mathcal{N}(s_i), PG(\mathcal{N}(s_i)))$. We use in the following PG as variable for the different proximity graphs $EMST$, RNG, GG, YG, and DT. $PG(\mathcal{N}(s_i))$ is the proximity graph on a set $\mathcal{N}(s_i)$ of nodes. All edges in $PG(\mathcal{N}(s_i))$ are bidirectional (Definition 4) according to the definitions of proximity graphs given in Section 4.1, but $T_{PG}(\mathcal{N}(s_i), PG(\mathcal{N}(s_i)))$ is not necessarily a subgraph of the communication graph $CG(\mathcal{S}, \mathcal{E})$.

The subgraph $T_{PG}(s_i, E_{PG}(s_i))$ which contains only edges originating at node s_i, $E_{PG}(s_i) = \{(s_i, s_j) \in T_{PG}(\mathcal{N}(s_i), PG(\mathcal{N}(s_i))) : \forall s_j \in \mathcal{N}(s_i)\}$, is the contribution of the node to the overall topology. All edges in $E_{PG}(s_i)$ are unidirectional and $T_{PG}(s_i, E_{PG}(s_i))$ is a subgraph of the communication graph $CG(\mathcal{S}, \mathcal{E})$, $E_{PG}(s_i) \subseteq \mathcal{E}$. The union of these contributions, abbreviated as $T_{PG}(S, E)$, is denoted by the term *localized computed*. $E = \bigcup_{s_i \in S} E_{PG}(s_i)$.

Definition 18 (localized computed topologies).

The localized computed topologies based on the Euclidean minimum spanning tree, Relative neighborhood graph, Gabriel graph, Yao graph and Delaunay triangulation are defined as:

$$T_{EMST}(S, E) = \bigcup_{s_i \in S} T_{EMST}(s_i, E_{EMST}(s_i)) \tag{5.1}$$
$$E_{EMST}(s_i) = \{(s_i, s_j) \in T_{EMST}(\mathcal{N}(s_i), EMST(\mathcal{N}(s_i))) : \forall s_j \in \mathcal{N}(s_i)\}$$

$$T_{RNG}(S, E) = \bigcup_{s_i \in S} T_{RNG}(s_i, E_{RNG}(s_i)) \tag{5.2}$$
$$E_{RNG}(s_i) = \{(s_i, s_j) \in T_{RNG}(\mathcal{N}(s_i), RNG(\mathcal{N}(s_i))) : \forall s_j \in \mathcal{N}(s_i)\}$$

$$T_{GG}(S, E) = \bigcup_{s_i \in S} T_{GG}(s_i, E_{GG}(s_i)) \tag{5.3}$$
$$E_{GG}(s_i) = \{(s_i, s_j) \in T_{GG}(\mathcal{N}(s_i), GG(\mathcal{N}(s_i))) : \forall s_j \in \mathcal{N}(s_i)\}$$

$$T_{YG_k}(S, E) = \bigcup_{s_i \in S} T_{YG_k}(s_i, E_{YG_k}(s_i)) \tag{5.4}$$
$$E_{YG_k}(s_i) = \{(s_i, s_j) \in T_{YG_k}(\mathcal{N}(s_i), YG_k(\mathcal{N}(s_i))) : \forall s_j \in \mathcal{N}(s_i)\}$$

$$T_{DT}(S, E) = \bigcup_{s_i \in S} T_{DT}(s_i, E_{DT}(s_i)) \tag{5.5}$$
$$E_{DT}(s_i) = \{(s_i, s_j) \in T_{DT}(\mathcal{N}(s_i), DT(\mathcal{N}(s_i))) : \forall s_j \in \mathcal{N}(s_i)\}$$

The topologies computed on the fully connected graph are identical to the proximity graphs defined in the previous chapter. These topologies are denoted with $EMST(S)$, $RNG(S)$, $GG(S)$, $YG_k(S)$ and $DT(S)$ to stay uniform with Definitions 9–13. We use the term *global* to distinguish these topologies from the localized computed topologies.

As mentioned in Section 4.1 $EMST(S)$, $YG_k(S)$, and $DT(S)$ may not be unique on a given set S of sites. Ties can be broken arbitrarily if the proximity graphs is computed centralized, but uniqueness is necessary for the distributed computation of these topologies. If there exist multiple edges with the same length, a deterministic rule is required to decide uniformly. Hence, each node s is assigned a unique $id(s)$ (such as an IP/MAC address) for breaking ties.

Definition 19 (uniform rule). *Given two edges (s_1, s_2) and (s_3, s_4), the deterministic rule is defined as*

$$\begin{aligned}
\|s_1, s_2\| > \|s_3, s_4\| \quad \Leftrightarrow \quad & \|s_1, s_2\| > \|s_3, s_4\| \\
\textit{or} \quad & (\|s_1, s_2\| = \|s_3, s_4\| \\
& \&\& \max\{id_{s_1}, id_{s_2}\} > \max\{id_{s_3}, id_{s_4}\}) \\
\textit{or} \quad & (\|s_1, s_2\| = \|s_3, s_4\| \\
& \&\& \max\{id_{s_1}, id_{s_2}\} = \max\{id_{s_3}, id_{s_4}\} \\
& \&\& \min\{id_{s_1}, id_{s_2}\} > \min\{id_{s_3}, id_{s_4}\}).
\end{aligned}$$

The localized computed topologies are important for our further work, although they are only abstract. In general, no device in the wireless ad-hoc network is aware of the whole $T_{PG}(S, E)$. Nevertheless, this graphs must satisfy the quality properties of topologies described in Chapter 3. We analyze in the following sections the presented localized computed topologies regarding connectivity, energy-efficiency, planarity, node degree, transmission-efficiency, and robustness with respect to mobility.

5.2 Connectivity

According to the principles defined in Chapter 3 connectivity is the most basic requirement of a topology. It is still unacceptable to loose connectivity in the topology as long as the communication graph is connected. Topology control algorithms that cannot guarantee this property are unsuitable for practical solutions.

The localized computed topologies $T_{EMST}(S, E)$, $T_{RNG}(S, E)$, $T_{GG}(S, E)$, $T_{YG_k}(S, E)$, and $T_{DT}(S, E)$ are obviously subgraphs of the communication graph

$CG(\mathcal{S}, \mathcal{E})$. The following Lemmata 12-17 prove the connectivity of these localized computed topologies. We assume that the communication ranges at all nodes are circular and that $CG(\mathcal{S}, \mathcal{E})$ is bi-connected.

Each node computes only is local contribution to the overall topology. Hence, if a node s_i includes an edge to a neighbor node s_j in its local topology, the edge is just unidirectional. Only if node s_j includes edge (s_j, s_i) in its local topology becomes the edge bidirectional (Definition 4).

5.2.1 Euclidean minimum spanning tree

The communication edges in $T_{EMST}(S, E)$ are not necessarily bidirectional, i.e., $(s_i, s_j) \in T_{EMST}(s_i, E_{EMST}(s_i)) \nRightarrow (s_j, s_i) \in T_{EMST}(s_j, E_{EMST}(s_j))$. Figure 5.1 gives such an example. The Euclidean spanning tree of $\mathcal{N}(s_1)$ contains the following edges $\{(s_1, s_2), (s_1, s_6)\}$. Since $T_{EMST}(\mathcal{N}(s_6), EMST(\mathcal{N}(s_6)))$ consists of $\{(s_1, s_2), (s_2, s_3), (s_3, s_4), (s_4, s_5), (s_5, s_6)\}$ the edge (s_1, s_6) is an unidirectional edge.

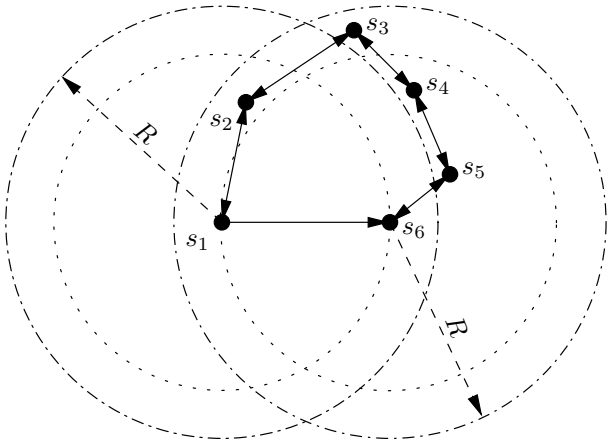


Figure 5.1: *Communication edges in $T_{EMST}(S, E)$ may be unidirectional.*

Hence, the localized computed Euclidean minimum spanning tree is not bi-connected, but we prove in the following lemma that $T_{EMST}(S, E)$ is strongly connected.

Lemma 12. *The localized computed Euclidean minimum spanning tree $T_{EMST}(S, E)$ is strongly connected.*

Proof. Suppose $CG(\mathcal{S}, \mathcal{E})$ is bi-connected and there are disconnected components $C_1, C_2, \ldots, C_M$ in $T_{EMST}(S, E)$. Let s_i and s_j be two nodes with $\|s_i, s_j\|$ is

minimal such that $(s_i, s_j) \in CG(\mathcal{S}, \mathcal{E})$ but $(s_i, s_j) \notin T_{EMST}(S, E)$ and $s_i \in C_i$, $s_j \in C_j$, $C_i \neq C_j$.

We prove in the following the connectivity in the neighborhood of node s_i, the proof is the same for the neighborhood of node s_j.

Since edge $(s_i, s_j) \notin T_{EMST}(S, E)$, there must be a path

$$\mathcal{P}(s_i, s_j) \in T_{EMST}(\mathcal{N}(s_i), EMST(\mathcal{N}(s_i))),$$

$$\mathcal{P}(s_i, s_j) = (s_i = t_0, t_1, \ldots, t_{k-1}, t_k = s_j),\ k \geq 2.$$

For each edge (t_j, t_{j+1}), $j = 0, 1, \ldots, k-1$, is $\|t_j, t_{j+1}\| < \|s_i, s_j\|$ and $t_{j+1} \in \mathcal{N}(t_j)$. There exists two cases:

- $\exists \mathcal{P}(t_j, t_{j+1}) \in T_{EMST}(\mathcal{N}(t_j), EMST(\mathcal{N}(t_j)))$, $0 \leq j < k$
  $\mathcal{P}(t_j, t_{j+1}) = (t_j = u_0, u_1, \ldots, u_{k-1}, u_k = t_{j+1})$, $k \geq 1$,
  $\Rightarrow$ contradiction to the demanded disconnectivity of the components.

- $\nexists \mathcal{P}(t_j, t_{j+1}) \in T_{EMST}(\mathcal{N}(t_j), EMST(\mathcal{N}(t_j)))$ violates the minimality assumption for edge (s_i, s_j).

□

The localized computed Euclidean minimum spanning tree differs considerably from the global Euclidean minimum spanning tree. It is neither minimal nor a spanning tree. $T_{EMST}(S, E)$ contains many more edges than $EMST(S)$ and does not achieve the bi-connectivity of $EMST(S)$. However, Li et al. [LHS05] show that $T_{EMST}(S, E)$ is bi-connected if all unidirectional edges are removed. The localized computed Euclidean minimum spanning tree is a supergraph of $EMST(S)$ if $CG(\mathcal{S}, \mathcal{E})$ is connected and if the nodes use homogeneous transmission ranges.

5.2.2 Relative neighborhood graph and Gabriel graph

The localized computed Relative neighborhood graph as well as the localized computed Gabriel graph, are bi-connected. The critical section of an edge in Relative neighborhood graph and Gabriel graph are complete within the circular and homogeneous communication range of the participating nodes. Hence, both respective nodes of an edge make the same decision.

Lemma 13. *The localized computed Relative neighborhood graph $T_{RNG}(S, E)$ preserves the bi-connectivity of $CG(\mathcal{S}, \mathcal{E})$.*

Proof. Suppose $CG(\mathcal{S}, \mathcal{E})$ is bi-connected and there are disconnected components $C_1, C_2, \ldots, C_M$ in $T_{RNG}(S, E)$. Let s_i and s_j be two nodes with $\|s_i, s_j\|$ is minimal such that $(s_i, s_j) \in CG(\mathcal{S}, \mathcal{E})$ but $(s_i, s_j) \notin T_{RNG}(S, E)$ and $s_i \in C_i$, $s_j \in C_j$, $C_i \neq C_j$.

Since edge $(s_i, s_j) \in CG(\mathcal{S}, \mathcal{E})$, there must be a node s_k in the critical section (the shaded regions in Figure 4.2(a)) of s_i and s_j. Since $\|s_i, s_k\| < \|s_i, s_j\|$ and $\|s_j, s_k\| < \|s_i, s_j\|$, $(s_i, s_k) \in CG(\mathcal{S}, \mathcal{E})$ and $(s_j, s_k) \in CG(\mathcal{S}, \mathcal{E})$. There are three cases:

- (s_i, s_k), $(s_j, s_k) \notin T_{RNG}(S, E)$
  $\Rightarrow \exists s_k$ with $s_k \in C_k$, $C_k \neq C_i$, $C_k \neq C_j$ and $\|s_i, s_k\| < \|s_i, s_j\|$ and $\|s_j, s_k\| < \|s_i, s_j\|$, (s_i, s_j) violates minimality assumption.

- (s_i, s_k) or $(s_j, s_k) \notin T_{RNG}(S, E)$
  $\Rightarrow$ since $\|s_i, s_k\| < \|s_i, s_j\|$ and $\|s_j, s_k\| < \|s_i, s_j\|$, (s_i, s_j) violates minimality assumption.

- Both (s_i, s_k), $(s_j, s_k) \in T_{RNG}(S, E)$
  $\Rightarrow$ contradiction to the demanded disconnectivity of the components.

□

Lemma 14. *The localized computed Gabriel graph $T_{GG}(S, E)$ preserves the connectivity of $CG(\mathcal{S}, \mathcal{E})$.*

Proof. The proof for $T_{GG}(S, E)$ is the same as for $T_{RNG}(S, E)$. Only the critical section of an edge in the Gabriel graph is a circle (see Figure 4.3(a)) and not a lune as in the Relative neighborhood graph. □

5.2.3 Yao graph

Lemma 15. *The localized computed Yao graph $T_{YG_k}(S, E)$ preserves the bi-connectivity of $CG(\mathcal{S}, \mathcal{E})$, if $k \geq 6$ in $T_{YG_k}(s, E_{YG_k}(s))$, $\forall s \in S$*

Proof. Suppose $CG(\mathcal{S}, \mathcal{E})$ is bi-connected and there are disconnected components $C_1, C_2, \ldots, C_M$ in $T_{YG_k}(S, E)$. Let s_i and s_j be two nodes with $\|s_i, s_j\|$ is minimal such that $(s_i, s_j) \in CG(\mathcal{S}, \mathcal{E})$ but $(s_i, s_j) \notin T_{YG_k}(S, E)$ and $s_i \in C_i$, $s_j \in C_j$.

Since edge $(s_i, s_j) \in CG(\mathcal{S}, \mathcal{E})$, there must be a node s_l in $cone(s_i, s_j)$. Since $\|s_i, s_l\| < \|s_i, s_j\|$, we have $\|s_j, s_l\| < \|s_i, s_j\|$ because $\Theta \leq \Pi/3$ (Lemma 1) and hence, $(s_j, s_l) \in CG(\mathcal{S}, \mathcal{E})$. There are three cases:

- (s_i, s_l), $(s_j, s_l) \notin T_{YG_k}(S, E) \Rightarrow \exists s_l$ with $s_l \in C_l$, $C_l \neq C_i$, $C_l \neq C_j$ and $\|s_i, s_l\| < \|s_i, s_l\|$ and $\|s_j, s_l\| < \|s_i, s_j\|$, (s_i, s_j) violates minimality assumption.

- (s_i, s_l) or $(s_j, s_l) \notin T_{YG_k}(S, E) \Rightarrow$ since $\|s_i, s_l\| < \|s_i, s_j\|$ and $\|s_j, s_l\| < \|s_i, s_j\|$, (s_i, s_j) violates minimality assumption.

- Both (s_i, s_l), $(s_j, s_l) \in T_{YG_k}(S, E) \Rightarrow$ contradiction to the demanded disconnectivity of the components.

□

5.2.4 Delaunay triangulation

The local Delaunay triangulation $T_{DT}(\mathcal{N}(s), DT(\mathcal{N}(s)))$, $s \in S$, at each node is only computed by considering the single hop neighborhood. A node s uses only the positions of the nodes within its communication range for the computation of $T_{DT}(\mathcal{N}(s), DT(\mathcal{N}(s)))$ and ignores the nodes farther away than the communication range. However, information of the single hop neighborhood may not be enough to review the complete critical section of a Delaunay edge. This critical section is shown in Figure 4.5(a). The local Delaunay triangulations at different nodes might be inconsistent, i.e., a communication edge may be in the local Delaunay triangulation $T_{DT}(\mathcal{N}(s_i), DT(\mathcal{N}(s_i)))$ of node s_i but not in $T_{DT}(\mathcal{N}(s_j), DT(\mathcal{N}(s_j)))$. Hence, $T_{DT}(S, E)$ is not bi-connected but strongly connected.

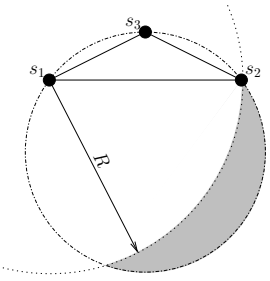


Figure 5.2: *The shaded region is outside of s_1's communication range.*

Figure 5.2 explain this issue: If the triangle (s_1, s_2, s_3) is a Delaunay Triangle, then no other node is inside the circumcircle of this triangle by Definition 13. Node s_1 cannot communicate with a node in the shaded region, so for node s_1 the edge (s_1, s_2) is a Delaunay edge. If a node was located in the shaded region and within the communication range of node s_2, then node s_2 would omit the edge to s_1, because the empty circle property (Definition 14) is violated for edge (s_1, s_2). Hence, edge (s_1, s_2) would be unidirectional.

The localized computed Delaunay triangulation is, by Definition 18, the union of the local Delaunay triangulations $DT(s), \forall s \in S$. For each edge (s_i, s_j) of the communication graph $CG(\mathcal{S}, \mathcal{E})$ that is not in $T_{DT}(S, E)$, the edge is neither in $T_{DT}(s_i, E_{DT}(s_i)))$ nor in $T_{DT}(s_j, E_{DT}(s_j))$. Before we prove the connectivity of $T_{DT}(S, E)$ we need a lemma to verify the conditions when an edge is not a Delaunay edge.

Lemma 16. *An edge (s_i, s_j) of $CG(\mathcal{S}, \mathcal{E})$ is not in $T_{DT}(s_i, E_{DT}(s_i))$, if and only if (s_i, s_j) is intersected by a Delaunay edge (s_k, s_l). s_i, s_j, s_k, $s_l \in \mathcal{N}(s_i)$.*

Proof. By Lemma 2, the local Delaunay triangulation $T_{DT}(\mathcal{N}(s_i), DT(\mathcal{N}(s_i)))$ is planar. It is by Lemma 3 impossible to add an edge to a Delaunay triangulation without destroying the planarity. Hence, each non-Delaunay edge is intersected by at least one Delaunay edge. □

Lemma 17. *The localized computed Delaunay triangulation $T_{DT}(S, E)$ is strongly connected.*

Proof. Suppose $CG(\mathcal{S}, \mathcal{E})$ is bi-connected and there are disconnected components $C_1, C_2, \ldots, C_M$ in $T_{DT}(S, E)$. Let s_i and s_j be two nodes with $\|s_i, s_j\|$ is minimal such that $(s_i, s_j) \in CG(\mathcal{S}, \mathcal{E})$ but $(s_i, s_j) \notin T_{DT}(S, E)$ and $s_i \in C_i$, $s_j \in C_j$.

We prove in the following the connectivity in the neighborhood of node s_i, the proof is the same for the neighborhood of node s_j.

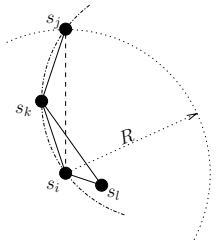


Figure 5.3: *Edge (s_i,s_j) is no Delaunay edge*

Since edge $(s_i, s_j) \in CG(\mathcal{S}, \mathcal{E})$, there must be some node s_l inside the circumcircle of the triangle (s_i, s_j, s_k) (Figure 5.3, Definition 13). Edge (s_i, s_j) divides the circle through s_i and s_j in two circle segments, whereas node s_k lies in the first segment and node s_l lies in the other segment (Lemma 16). In at least one segment, w.l.o.g. the segment with node s_k, $\|s_i, s_k\| < \|s_i, s_j\|$ and $\|s_j, s_k\| < \|s_i, s_j\|$, $(s_i, s_k) \in CG(\mathcal{S}, \mathcal{E})$ and $(s_j, s_k) \in CG(\mathcal{S}, \mathcal{E})$. There are three cases:

- (s_i, s_k), $(s_j, s_k) \notin T_{DT}(S, E) \Rightarrow \exists s_k$ with $s_k \in C_k$, $C_k \neq C_i$, $C_k \neq C_j$ and $\|s_i, s_k\| < \|s_i, s_j\|$ and $\|s_j, s_k\| < \|s_i, s_j\|$, (s_i, s_j) violates minimality assumption.

- (s_i, s_k) or $(s_j, s_k) \notin T_{DT}(S, E) \Rightarrow$ since $\|s_i, s_k\| < \|s_i, s_j\|$ and $\|s_j, s_k\| < \|s_i, s_j\|$, (s_i, s_j) violates minimality assumption.

- Both (s_i, s_k),$(s_j, s_k) \in T_{DT}(S, E) \Rightarrow$ contradiction to the demanded disconnectivity of the components.

□

Some communication edges in the localized computed Delaunay triangulation may be unidirectional. Unidirectional communication edges are a grave drawback for routing algorithms. Marina and Das [MD02] have shown that the high overhead needed to handle unidirectional edges in routing protocols outweighs the benefits that they can provide, and better performance can be achieved by simply avoiding them. Chapter 6 shows that simple communication between single hop neighboring nodes helps to obtain bidirectional communication edges.

5.3 Energy-efficiency

5.3.1 Length Stretch Factor

Localized computed topologies inherit some properties of the proximity graphs. In computational geometry the *length stretch factor* of a proximity graph is defined as the maximum ratio of the length of the shortest path connecting any pair of sites in the graph to their Euclidean distance (Definition 15). A graph is called a *spanner*, if the spanning ratio is bounded by a constant. The Euclidean minimum spanning tree, the Relative neighborhood graph, and the Gabriel graph are not spanners for the fully connected graph, because their spanning ratios are unbounded. In sharp contrast, the Delaunay triangulation and the Yao graph have a constant spanning ratio.

The length stretch factors of the proximity graphs (Lemmata 4-8) do not apply to the localized computed topologies, because those topologies do not contain arbitrarily long edges. The length stretch factor as defined in Definition 15 is only valid for topologies computed for the fully connected network. In all sparser networks the length stretch factor as defined in Definition 15 is inapplicable.

However, the length stretch factor used in computational geometry can be generalized for the usage in position-based topology control. The length stretch factor

of a proximity graph is defined as the maximum ratio of the shortest path length connecting any pair of sites in G with respect to their Euclidean distance. Now, we define the length stretch factor as the maximum ratio of the shortest path length connecting any pair of sites in the localized computed topology with respect to the shortest path length connecting this pair of sites in the communication graph. If the communication graph is the fully connected graph so the definition of the new length stretch factor is the same as the definition of the old length stretch factor. The length of the shortest path connecting s_i and s_j in graph G measured by Euclidean distance is denoted by $\Pi_G(s_i, s_j)$.

Definition 20 (Length stretch factor). *The length stretch factor of the localized computed topologies with respect to the communication graph is defined by:*

$$\sigma_{T_{PG}} := \max_{(s_i,s_j)} \frac{\Pi_{T_{PG}}(s_i, s_j)}{\Pi_{CG}(s_i, s_j)} \qquad \forall s_i, s_j \in S \tag{5.6}$$

Definition 21 (Spanner). *A localized computed topology T_{PG} is called a spanner, if $\Pi_{T_{PG}}(s_i, s_j)$ is no more than a constant factor larger than $\Pi_{CG}(s_i, s_j)$, for all $s_i, s_j \in S$.*

Lemma 20 will show that the length stretch factor of T_{PG} is at most the length stretch factor of $PG(S)$.

We show that the length of the shortest path connecting two nodes in a localized computed topology is at most the length stretch factor of the proximity graph times the length of the shortest path connecting this two nodes in the communication graph. Each edge (s_i, s_j) of the shortest path in the communication graph is either part of the localized computed topology or there must be at least one path $\mathcal{P}(s_i, s_j)$ connecting s_i and s_j in the localized computed topology. We show that the shortest path $\mathcal{P}(s_i, s_j)$ connecting s_i and s_j in $T_{PG}(S, E)$ is exactly the same as the shortest path connecting these two nodes in the global topology $PG(S)$. Hence, the ratios between the paths and the Euclidean distance $\|s_i, s_j\|$ must also be the same. This is obvious if the communication graph is fully connected, because $T_{PG}(S, E) = PG(S)$ in a fully connected network. However, for the localized computed topology we have to show that the shortest path really exists, i.e., the edges in the shortest path must be shorter than the communication range (Lemma 18) and each edge of the global topology that is shorter than the communication range must be present in the localized computed topology (Lemma 19).

The following Lemma 18 proves a stronger result:

Lemma 18. *The edge(s) of the shortest path connecting two sites s_i and s_j in the localized computed topology are shorter or equal than $\|s_i, s_j\|$, $(s_i, s_j) \in CG(\mathcal{S}, \mathcal{E})$.*

Proof.

Euclidean minimum spanning tree.
An edge (s_i, s_j) is not included in $T_{EMST}(S, E)$ only if the edges in the path connecting s_i and s_j in $T_{EMST}(\mathcal{N}(s_i), EMST(\mathcal{N}(s_i)))$ are shorter than $\|s_i, s_j\|$.

Relative neighborhood graph.
Suppose an edge $(s_i, s_j) \in CG(\mathcal{S}, \mathcal{E})$ with $\|s_i, s_j\|$ is minimal and $(s_i, s_j) \notin T_{RNG}(S, E)$. Since edge $(s_i, s_j) \in CG(\mathcal{S}, \mathcal{E})$, there must be a site s_k in the critical section of (s_i, s_j). Assume for contradiction that either $\|s_i, s_k\| > \|s_i, s_j\|$, $\|s_j, s_k\| > \|s_i, s_j\|$, or both. Consider a hypothetical point p in the critical section of (s_i, s_j). No matter where p is, $\|s_i, p\| < \|s_i, s_j\|$ and $\|s_j, p\| < \|s_i, s_j\|$. This is a contradiction to the existence of s_k in the critical section of (s_i, s_j).

Gabriel graph.
The proof for $T_{GG}(S, E)$ is similar to the proof for $T_{RNG}(S, E)$.

Yao graph.
The proof for $T_{YG_k}(S, E)$ is similar to the proof for $T_{RNG}(S, E)$.

Delaunay triangulation.
Suppose an edge $(s_i, s_j) \in CG(\mathcal{S}, \mathcal{E})$ with $\|s_i, s_j\|$ is minimal and $(s_i, s_j) \notin T_{DT}(S, E)$. Let C be a circle whose boundary passes through s_i and s_j, with the diameter $\|s_i, s_j\|$. Since edge $(s_i, s_j) \in CG(\mathcal{S}, \mathcal{E})$, there lies by Definition 14 at least one site s_k within circle C and at least one site lies in circle C_1 which passes through s_i, s_j, and s_k. At least $\|s_i, s_k\| < \|s_i, s_j\|$ and $\|s_j, s_k\| < \|s_i, s_j\|$.

□

Lemma 19. *The localized computed topologies contain all edges of the global topologies that are shorter than the communication range.*

Proof. For the sake of contradiction, we assume an edge $(s_i, s_j) \in CG(\mathcal{S}, \mathcal{E})$, $(s_i, s_j) \notin T_{PG}(S, E)$ and $(s_i, s_j) \in PG(S)$.

An edge is not in $T_{RNG}(S, E)$, $T_{GG}(S, E)$, $T_{YG_k}(S, E)$, or $T_{DT}(S, E)$ if there is a site s_k in the critical section of edge (s_i, s_j). However, s_k is also part of $RNG(S)$, $GG(S)$, $YG_k(S)$, and $DT(S)$. Hence, $(s_i, s_j) \notin PG(S)$.

Edge $(s_i, s_j) \notin T_{EMST}(S, E)$ only if there exists a tree of minimal total edge length in $EMST(N(s_i))$ connecting all sites in $N(s_i)$ without edge (s_i, s_j). Obviously, the same tree is also available for $EMST(S)$, so $(s_i, s_j) \notin EMST(S)$. □

Hence, each edge in the communication graph is either part of the localized computed topology or there exists a shortest path which is only the length stretch factor of the proximity graph larger than the edge in the communication graph.

Lemma 20. *The length of the shortest path in a localized computed topology $T_{PG}(S, E)$ is at most the length stretch factor of the proximity graph times the shortest path in the communication graph.*
The length stretch factor of $T_{PG}(S, E)$ is:

$$\sigma_{T_{PG}} \leq \sigma_{PG} \tag{5.7}$$

Proof. The shortest path connecting node s_i and node s_j in the communication graph $CG(\mathcal{S}, \mathcal{E})$ consists of m edges:

$$\begin{aligned} \Pi_{CG}(s_i, s_j) &= \|s_i = t_0, t_1\| + \|t_1, t_2\| + \dots \\ &\quad \dots + \|t_{m-1}, t_m = s_j\| \end{aligned} \tag{5.8}$$

Due to Lemma 18 and Definition 20, $\|t_l, t_{l+1}\| \geq \frac{\Pi_{T_{PG}}(t_l, t_l+1)}{\sigma_{PG}}$, $0 \leq l < m$.

$$\begin{aligned} \Pi_{CG}(s_i, s_j) &\geq \frac{\Pi_{T_{PG}}(s_i = t_0, t_1)}{\sigma_{PG}} + \frac{\Pi_{T_{PG}}(t_1, t_2)}{\sigma_{PG}} + \dots \\ &\quad \dots + \frac{\Pi_{T_{PG}}(t_{m-1}, t_m = s_j)}{\sigma_{PG}} \end{aligned} \tag{5.9}$$

$$\begin{aligned} \Pi_{T_{PG}}(s_i, s_j) &= \Pi_{T_{PG}}(s_i = t_0, t_1) + \Pi_{T_{PG}}(t_1, t_2) + \dots \\ &\quad \dots + \Pi_{T_{PG}}(t_{m-1}, t_m = s_j) \end{aligned} \tag{5.10}$$

$$\Rightarrow \frac{\Pi_{T_{PG}}(s_i, s_j)}{\Pi_{CG}(s_i, s_j)} \leq \sigma_{PG} \tag{5.11}$$

$$\Rightarrow \sigma_{T_{PG}} \leq \sigma_{PG} \tag{5.12}$$

□

5.3.2 Power Stretch Factor

It is known [Rap01] that the power $Pow(s_i, s_j)$ required by node s_i to correctly transmit data to node s_j must satisfy inequality

$$\frac{Pow(s_i, s_j)}{\|s_i, s_j\|^\alpha} \geq \beta \tag{5.13}$$

where $\alpha \geq 2$ is the *distance-power gradient* and $\beta \geq 1$ is the *transmission quality* parameter. While the value of β is usually set to 1, the value of α depends on environmental conditions. In the ideal case, we have $\alpha = 2$; however, α is typically 4 in realistic situations. A value of α in the interval $[2, 6]$ is commonly accepted.

Inequality (5.13) accounts only for the power consumed by the sender node (transmit power). In practice, in radio communication a non-negligible amount of energy is consumed at the receiver node to receive and decode the transmitted signal. Most of the current literature does not account for the receiver energy, and the design of more realistic energy models is one of the main open issues in the field. Formula (5.13) holds for free-space environments with non-obstructed line of sight, and it does not consider the possible occurrence of reflections, scattering and diffractions caused by buildings, terrain, and so on. Although more complicated formulae of the radio signal attenuation with distance are known, such as that derived in [FBS02], inequality (5.13) is widely accepted in the ad-hoc network community.

Let $CG(\mathcal{S}, \mathcal{E})$ be the communication graph obtained when all the nodes transmit at the same, maximum power and assume $CG(\mathcal{S}, \mathcal{E})$ is connected. Every edge (s_i, s_j) in $CG(\mathcal{S}, \mathcal{E})$ is weighted with the power $\|s_i, s_j\|^\alpha$ needed to transmit a message between s_i and s_j. Given any path $\mathcal{P}(s_i, s_j) = (s_i = t_0, t_1 \ldots, t_k = s_j)$, the power cost $Pow(\mathcal{P}(s_i, s_j))$ of $\mathcal{P}(s_i, s_j)$ is defined as the sum of the power costs of the single edges, i.e., $Pow(\mathcal{P}(s_i, s_j)) = \sum_{i=0}^{k-1} \|t_i, t_{i+1}\|^\alpha$.

Definition 22 (Minimum power path). *Let $\Phi_T(s_i, s_j)$ denote the minimum of $Pow(\mathcal{P}(s_i, s_j))$ over all paths $\mathcal{P}(s_i, s_j)$ that connect nodes s_i and s_j in topology T. A path in T connecting s_i and s_j and consuming the minimum power is called the minimum-power path between s_i and s_j.*

Note, the minimum power path s_i and s_j is not necessarily identical to the shortest path connecting s_i and s_j!

Definition 23 (Power stretch factor). *The localized computed topology $T_{PG}(S, E)$ is a subgraph of $CG(S)$. The power stretch factor of $T_{PG}(S, E)$ with respect to*

$CG(S)$ is the maximum over all possible node pairs of the ratio between the cost of the minimum-power paths in $T_{PG}(S, E)$ and in $CG(S)$. Formally,

$$\mu_{T_{PG}} := \max_{(s_i,s_j)} \frac{\Phi_{T_{PG}}(s_i, s_j)}{\Phi_{CG}(s_i, s_j)} \qquad \forall s_i, s_j \in S.$$

The power stretch factor is a generalization of the concept of length stretch factor. If we define $\alpha = 1$, we obtain the length stretch factor as given in Definition 20.

Lemma 21. *For a localized computed topology with length stretch factor $\sigma_{T_{PG}}$ the power stretch factor $\mu_{T_{PG}}$ is at most $\sigma^{\alpha}_{T_{PG}}$.*

Proof. Let $\Phi_{CG}(s_i, s_j) = (s_i = t_0, t_1 \ldots, t_l = s_j)$ be the minimum power path connecting two nodes s_i and s_j in the communication graph $CG(\mathcal{S}, \mathcal{E})$. For each edge $(t_i, t_{i+1}) \in CG(\mathcal{S}, \mathcal{E})$ exists a shortest path $\mathcal{P}(t_i, t_{i+1})$ in T_{PG}

$$\mathcal{P}(t_i, t_{i+1}) = (t_i = u_0, u_1 \ldots, u_m = t_{i+1}),\ m \geq 1, \tag{5.14}$$

$$\Pi_{T_{PG}}(t_i, t_{i+1}) \leq \sigma_{T_{PG}} \|t_i, t_{i+1}\|. \tag{5.15}$$

$$\begin{aligned} Pow(\mathcal{P}(t_i, t_{i+1})) \;&=\; \sum_{j=0}^{m-1} \|u_j, u_{j+1}\|^{\alpha} \leq (\sum_{j=0}^{m-1} \|u_j, u_{j+1}\|)^{\alpha} \leq \\ &\quad \sigma^{\alpha}_{T_{PG}} \|t_i, t_{i+1}\|^{\alpha}. \end{aligned} \tag{5.16}$$

For the whole path,

$$\begin{aligned} Pow(\mathcal{P}(s_i, s_j)) \;&\leq\; \sum_{k=0}^{l-1} Pow(\mathcal{P}(t_k, t_{k+1})) \leq \sigma^{\alpha}_{T_{PG}} \sum_{k=0}^{l-1} \|t_k, t_{k+1}\|^{\alpha} = \\ &\quad \sigma^{\alpha}_{T_{PG}} \Phi_{CG}(s_i, s_j). \end{aligned} \tag{5.17}$$

Hence, $\mu_{T_{PG}} \leq \sigma^{\alpha}_{T_{PG}}$.

□

A spanner, as defined in Definition 21, is a subgraph of $CG(\mathcal{S}, \mathcal{E})$ in which the length of the shortest path connecting any two nodes is within a constant factor of the length of the shortest path connecting the two nodes in $CG(\mathcal{S}, \mathcal{E})$. The length stretch factor of a spanner is $O(1)$. If we adopt the concept of a spanner for the power consumption setting, setting then a topology with a $O(1)$ length stretch factor also has a $O(1)$ power stretch factor (Lemma 21). The reverse implication

is not true, however: The required transmission power increases with exponent α as distance is raised, so short communication links are more energy efficient than long communication links. The localized computed topologies $T_{YG_k}(S, E)$, $k > 6$, and $T_{DT}(S, E)$ are power spanners of the communication graph.

The bound of the power stretch factor given in Lemma 21 are very imprecise. The estimate in Equation 5.16 is very rough and the path used in $T_{PG}(S, E)$ is not necessarily the minimal power path. The effective power stretch factors will be lower. The following list gives the power stretch factors of the localized computed topologies presented in Definition 18:

Euclidean minimum spanning tree and Relative neighborhood graph.
The length stretch factors of the Euclidean minimum spanning tree and the Relative neighborhood graph is $N - 1$, $N = |S|$. Lemma 21 implies that the power stretch factor is at most $(N - 1)^\alpha$. However, the power stretch factor of $T_{EMST}(S, E)$ and $T_{RNG}(S, E)$ is $N - 1$ [LWW01].

Therefore, any graph that contains the Euclidean minimum spanning tree (e.g., Gabriel graph, Yao graph, Delaunay triangulation) has the power stretch factor at most $N - 1$.

Gabriel graph.
Since the Gabriel graph has length stretch factor $\sqrt{N-1}$, than Lemma 21 implies that its power stretch factor is at most $(\sqrt{N-1})^\alpha$. The following lemma shows that the localized computed Gabriel graph has power stretch factor $\mu_{T_{GG}} = 1$, for $\alpha > 2$. Hence, $T_{GG}(S, E)$ is a power spanner of the communication graph.

Lemma 22. *The minimal power path connecting two nodes in $CG(\mathcal{S}, \mathcal{E})$ is identically to the minimal power path connecting the two nodes $T_{GG}(S, E)$.*

Proof. Let (s_i, s_j) be the shortest edge of the minimal power path connecting two nodes in $CG(\mathcal{S}, \mathcal{E})$ and $(s_i, s_j) \notin T_{GG}(S, E)$. An edge is not in $T_{GG}(S, E)$, if there lies a node s_k inside or on the circle drawn through s_i and s_j with diameter $\|s_i, s_j\|$. For $\alpha > 2$, we have $\|s_i, s_k\|^\alpha + \|s_j, s_k\|^\alpha < \|s_i, s_j\|^\alpha$. This contradicts the existence of (s_i, s_j) in the minimal power path. □

Yao graph.
The localized computed Yao graph $T_{YG_k}(S, E)$ has length stretch factor $\frac{1}{1-2\sin\frac{\pi}{k}}$ for $k > 6$. Lemma 21 implies its power stretch factor is no more

than $(\frac{1}{1-2\sin\frac{\pi}{k}})^{\alpha}$. Li et al. [LWW01] show that the power stretch factor $\mu_{T_{YG_k}}$ of $T_{YG_k}(S,E)$ is at most $\frac{1}{1-(2\sin\frac{\pi}{k})^{\alpha}}$.

Delaunay triangulation.
The power stretch factor of the localized computed Delaunay triangulation is due to Lemma 21 at most $(\frac{2\pi}{3\cos(\frac{\pi}{6})})^{\alpha}$. However, $\mu_{T_{DT}} = \mu_{T_{GG}}$, because $T_{GG}(S,E) \subseteq T_{DT}(S,E)$.

The localized computed Gabriel graph and the localized computed Delaunay triangulation are energy-optimal, since they have a power stretch factor of 1.

5.4 Planarity

Planarity is an important property for wireless ad-hoc networks in the context of routing. Some routing algorithms require a planar topology to facilitate guaranteed delivery of messages. As mentioned in Section 4.2.2, Euclidean minimum spanning tree, Relative neighborhood graph, Gabriel graph, and Delaunay triangulation are planar, but the Yao graph is non-planar. The Delaunay triangulation is furthermore not only planar but also one of the densest planar graphs. We verify in this section whether the planarity of the global topologies is inherited to the localized computed topologies.

5.4.1 Euclidean minimum spanning tree

We show in the following Lemma that $T_{EMST}(S,E)$ is a subgraph of $T_{GG}(S,E)$[1]. The planarity of $T_{GG}(S,E)$ is proved in Section 5.4.2.
The planarity of $T_{EMST}(S,E)$ follows immediately.

Lemma 23. $T_{EMST}(S,E) \subseteq T_{GG}(S,E)$

Proof. For the sake of contradiction, we assume that there is an edge (s_i,s_j) in $T_{EMST}(S,E)$ that is not in $T_{GG}(S,E)$. Hence,

$$(s_i,s_j) \in T_{EMST}(\mathcal{N}(s_i), EMST(\mathcal{N}(s_i))) \text{ and}$$
$$(s_i,s_j) \notin T_{GG}(\mathcal{N}(s_i), GG(\mathcal{N}(s_i))).$$

However, according to Lemma 1,

$$T_{EMST}(\mathcal{N}(s_i), EMST(\mathcal{N}(s_i))) \subseteq T_{GG}(\mathcal{N}(s_i), GG(\mathcal{N}(s_i))).$$

This contradicts the existence of (s_i,s_j) in $EMST(S)$. □

[1]The same result appears if we show $T_{EMST}(S,E) \subseteq T_{RNG}(S,E)$ instead of $T_{EMST}(S,E) \subseteq T_{GG}(S,E)$.

5.4.2 Relative neighborhood graph and Gabriel graph

The localized computed topologies $T_{RNG}(S, E)$ and $T_{GG}(S, E)$ contain all edges of their respective global topologies $RNG(S)$ and $GG(S)$ that are shorter than the communication range. A node s only needs information about its single hop neighborhood to decide whether a communication edge (s, t) exists in its local contribution to the overall topology ($T_{RNG}(s, E_{RNG}(s))$ resp. $T_{GG}(s, E_{GG}(s))$) or not, because the node can completely survey the critical section of (s, t). We show in the following Lemma 24 that the localized computed Relative neighborhood graph ($T_{RNG}(S, E)$) and the localized computed Gabriel graph ($T_{GG}(S, E)$) are subgraphs of their respective global topologies and hence planar.

Lemma 24. *$T_{RNG}(S, E) \subseteq RNG(S)$ and $T_{GG}(S, E) \subseteq GG(S)$*

Proof. A node s requires only the positions of the nodes in its neighborhood to decide whether an edge (s, s_i), $s_i \in S$, is valid in the $T_{RNG}(s, E_{RNG}(s))$ (resp. $T_{GG}(s, E_{GG}(s))$) or not. Hence, all edges that are present in only one local topology are also present in the global topology. The localized computed topology $T_{RNG}(S, E)$ (resp. $T_{GG}(S, E)$) consists of all edges of the global topology that are shorter than the communication range. If the communication range is infinity (i.e., the network is fully connected), then $T_{RNG}(S, E) = RNG(S)$ and $T_{GG}(S, E) = GG(S)$. □

Corollary 4. *$T_{EMST}(S, E)$, $T_{RNG}(S, E)$, and $T_{GG}(S, E)$ are planar.*

5.4.3 Yao graph

Obviously, $T_{YG_k}(S, E)$ violates the planarity requirement. Just as $T_{RNG}(S, E)$ and $T_{GG}(S, E)$, $T_{YG_k}(S, E)$ is a subgraph of the global graph. It is easy to see that the localized computed Yao graph contains all edges of the global Yao graph $YG(S)_k$ that are shorter than the communication range, but as $YG(S)_k$ is non-planar, so is $T_{YG_k}(S, E)$.

5.4.4 Delaunay triangulation

The localized computed Delaunay triangulation $T_{DT}(S, E)$ is not necessarily a subgraph of the global Delaunay triangulation. It contains all edges of the global Delaunay triangulation that are shorter than the communication range and some additional edges. Lemma 25 shows that $T_{DT}(S, E)$ might not be planar although the $T_{DT}(s, E_{DT}(s))$'s are planar individually.

Lemma 25. *The union of planar local Delaunay triangulations is not necessarily planar.*

$$\exists S \subset \mathbb{R}^2 \quad \textit{such that} \quad T_{DT}(S,E) \not\subseteq DT(S) \tag{5.18}$$

Proof. Figure 5.4 shows an example of 4 nodes ($S := \{s_1, s_2, s_3, s_4\}$) where $\bigcup_{i=1}^{4} T_{DT}(s_i, E_{DT}(s_i))$ is not a planar graph.

Figure 5.4(a) shows the communication graph of the network. The distance between node s_1 and node s_4 is longer than the communication range, the nodes s_1 and s_4 are therefore double hop neighbors. The local Delaunay triangulation $T_{DT}(\mathcal{N}(s_1), DT(\mathcal{N}(s_1)))$ of node s_1 and the edges originating at node s_1, i.e., $T_{DT}(s_1, E_{DT}(s_1))$ are shown in Figure 5.4(d). $T_{DT}(\mathcal{N}(s_2), DT(\mathcal{N}(s_2)))$ and $T_{DT}(\mathcal{N}(s_3), DT(\mathcal{N}(s_3)))$ are equal because both nodes can communicate directly with the same set of nodes. However, $T_{DT}(s_2, E_{DT}(s_2))$ and $T_{DT}(s_4, E_{DT}(s_4))$ are obviously different. These topologies are shown in Figure 5.4(e,f). Finally, $T_{DT}(\mathcal{N}(s_4), DT(\mathcal{N}(s_4))))$ and $T_{DT}(s_4, E_{DT}(s_4))$ are shown in Figure 5.4(g).

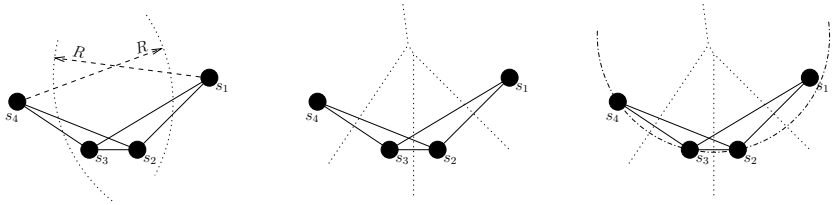
The localized computed Delaunay triangulation

$$T_{DT}(S,E) = \bigcup_{i=1}^{4} T_{DT}(s_i, E_{DT}(s_i))$$

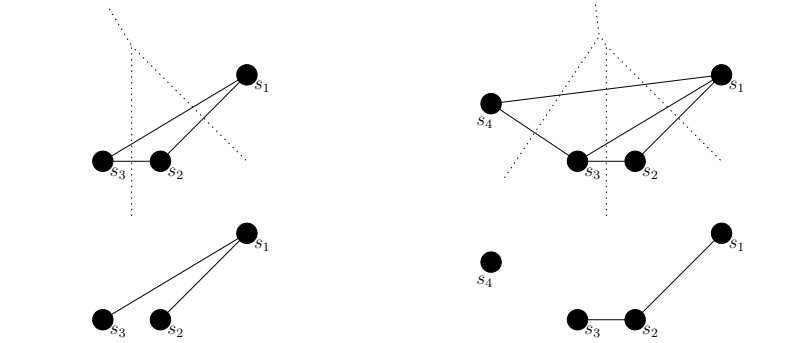
is not a planar graph because of the intersection of edge (s_1, s_3) with (s_2, s_4) (Figure 5.4(b)). Figure 5.4(c) shows that the edge (s_2, s_4) violates the conditions of a Delaunay Triangle, because the circumcircle of the triangle (s_2, s_3, s_4) contains node s_1 (Definition 13).

All local Delaunay triangulations $T_{DT}(\mathcal{N}(s_i), DT(\mathcal{N}(s_i)))$, $i = 1, \ldots, 4$ are planar individually. However, the localized computed Delaunay triangulation (shown in Figure 5.4(b)) is non planar, because edge (s_2, s_4) is unidirectional: Edge (s_2, s_4) belongs to $T_{DT}(s_4, E_{DT}(s_4))$, but not to $T_{DT}(s_2, E_{DT}(s_2))$. □

In contrast to $T_{RNG}(S,E)$ and $T_{GG}(S,E)$ the positions of the single hop neighbors are not sufficient to compute a subgraph of $DT(S)$. The localized computed Delaunay triangulation $T_{DT}(S,E)$ can contain edges which are not present in $DT(S)$ and so it may not fulfill the planarity property (Chapter 3). Hence, this topology is inapplicable for a lot of routing approaches which require planarity. Some additional computation is required to attain planarity. Chapter 6 proposes different approaches to this problem.

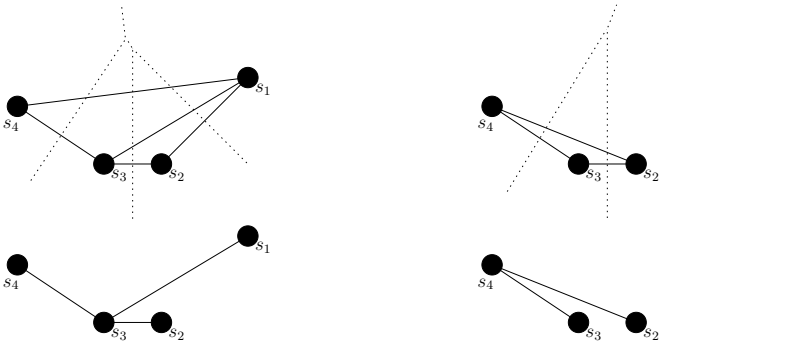


(a) $CG(\mathcal{S}, \mathcal{E})$ (b) $T_{DT}(S, E)$ (c) *non-delaunay edge* (s_2, s_4)



(d)
$T_{DT}(\mathcal{N}(s_1), DT(\mathcal{N}(s_1)))$,
$T_{DT}(s_1, E_{DT}(s_1))$

(e)
$T_{DT}(\mathcal{N}(s_2), DT(\mathcal{N}(s_2)))$,
$T_{DT}(s_2, E_{DT}(s_2))$



(f)
$T_{DT}(\mathcal{N}(s_3), DT(\mathcal{N}(s_3)))$,
$T_{DT}(s_3, E_{DT}(s_3))$

(g)
$T_{DT}(\mathcal{N}(s_4), DT(\mathcal{N}(s_4)))$,
$T_{DT}(s_4, E_{DT}(s_4))$

Figure 5.4: *local planar vs. global non planar*

5.5 Node degree, Transmission-efficiency, and Robustness to Mobility

The node degrees of the localized computed topologies are bounded by the degrees of the respective proximity graphs (see Section 4.2.5). Lemma 10 shows that 6 is the average node degree of $T_{EMST}(S, E)$, $T_{RNG}(S, E)$, $T_{GG}(S, E)$, and $T_{DT}(S, E)$. However, the maximum node degree is not constant in any of the considered graphs. For this reason, several variants of the above presented proximity graphs have been proposed, with the purpose of bounding the maximum node degree. Unfortunately, it has been shown that no proximity graph with constant node degree contains the minimum power path for every pair of nodes [WLF03]. Thus, no energy-optimal spanner with a constant bounded maximum node degree exists. To date, the routing graph with constant maximum node degree which has the best power stretch factor is the $OrdYaoGG$ graph of [SWLF04], which is obtained by building the $YG_k(S)$ graph, with $k > 6$, on top of the $GG(S)$. The $OrdYaoGG$ graph has power stretch factor of $\mu_{OrdYaoGG} = \frac{1}{1-(2\sin\frac{\pi}{k})^{\alpha}}$, and maximum node degree of $k + 5$, where $k > 6$ is the parameter of the Yao graph. For example, setting $k = 9$ and $\alpha = 2$ we have a power stretch factor of $1,88$ with a bound on the maximum node degree of 14.

Topologies based on proximity graphs communicate per definition only with the nearest neighbor nodes. The limitation of communication to nearest neighbor nodes reduces the interference in the wireless ad-hoc network and increases therefore the throughput of the network. A topology based on the Delaunay triangulation is the densest planar topology. All other position-based topologies are, by Lemma 1, subgraphs of this topology.

The computation of the topology has heavy effects on the robustness of a topology against mobility. The topologies presented in this chapter are totally localized. Each node uses only information of its single hop neighbor nodes for the computation of its contribution to the overall topology and no communication with nodes further away is required. This localized computation improves the robustness with respect to mobility. An important factor is also the computation algorithm for the local topology. In a position-based topology control algorithm each node s computes a local topology connecting the nodes of its neighborhood $\mathcal{N}(s)$. Nodes in a wireless ad-hoc network are generally not static; new nodes join the network, nodes leave the network (because of power constraints or errors) and mobile nodes change their positions. It would be very inefficient to recompute the proximity graph after each change in the wireless ad-hoc network. Successive maintenance of the proximity graph performs usually better. Appendix A gives an

overview of computation algorithms and examines the usability of the algorithms in dynamic environments.

5.6 Related Work

Bose and Morin [BM99] propose the global Delaunay triangulation as the topology. A topology identical to the global Delaunay triangulation is bi-connected but requires a fully connected network. The length of edges in the global Delaunay triangulation depends only on the geographic positions of the network nodes and are not bounded by any communication range. It is impossible to guarantee a fully connected network in an arbitrary wireless ad-hoc network.

Karp and Kung [KK00] propose two localized topology control algorithms for wireless ad-hoc networks. The first one is based on the Gabriel graph and the second one is based on the Relative neighborhood graph. Each node $s \in S$ computes its local contribution to the topology with the node positions of its neighborhood and generates a subgraph of its local contribution containing only edges originating at s. The union of these subgraphs is the localized computed topology, either based on the Gabriel graph or on the Relative neighborhood graph. The resulted topology is bi-connected and planar. Li and Hou [LH04] propose the same idea for topologies in heterogeneous networks (see Definition 6). They investigate topologies based on Euclidean minimum spanning trees and Relative neighborhood graphs. In heterogeneous networks the presented topologies are strongly connected. However, the communication ranges must be homogeneous to achieve bi-connected topologies.

Two methods for the localized computation of a topology based on the Delaunay triangulation are published in [LCWW03] and [GGH$^+$01b]. The algorithm of Gao et al. [GGH$^+$01b] presents a topology called *Restricted Delaunay Graph (RDG)* and Li et al. [LCWW03] propose the *PLDel(S)*. The topologies built by the methods presented in these papers are not only bi-connected and planar, but have also a better length stretch factor than the topologies based on Euclidean minimum spanning tree, Relative neighborhood graph, Gabriel graph, or Yao graph. A detailed description of these two topology control algorithms is given in Section 6.2.

Wage ruhig einen großen Sprung.
Über einen Abgrund kommt man
nicht mit zwei kleinen Schritten.

David Lloyd George (1863–1945)

Chapter 6

Short Delaunay Triangulation

The localized computed topologies presented in the previous chapter satisfy more or less the quality properties of topologies described in Chapter 3. These properties are connectivity, energy-efficiency, planarity, node degree, transmission-efficiency, and robustness to mobility. Table 6.1 summarizes different values and bounds describing the localized computed topologies presented in the previous chapter.

	$T_{EMST}(S,E)$	$T_{RNG}(S,E)$	$T_{GG}(S,E)$	$T_{YG_k}(S,E)$	$T_{DT}(S,E)$
connect.	strong con.	bi-con.	bi-con.	bi-con.	strong con.
σ	$N-1$	$N-1$	$\sqrt{N-1}$	$\frac{1}{1-2\sin\frac{\pi}{k}}$	$\frac{2\pi}{3\cos(\frac{\pi}{6})}$
μ	$N-1$	$N-1$	1	$\frac{1}{1-(2\sin\frac{\pi}{k})^{\alpha}}$	1
planar.	yes	yes	yes	no	no
aver.ND	$O(1)$	6	6	$O(1)$	6
max.ND	6	$N-1$	$N-1$	$N-1$	$N-1$

Abbreviations.

connect.	... connectivity	strong con.	... strong connectivity
bi.con.	... bi-connectivity	σ	... length stretch factor
μ	... power stretch factor	planar.	... planarity
aver.ND	... average node degree	max.ND	... maximum node degree

Figure 6.1: Connectivity, spanning factors, planarity, and node degrees of different localized computed topologies.

The localized computed Yao graph and the localized computed Delaunay triangulation are the only topologies with a constant length stretch factor. A constant power stretch factor is more important in wireless ad-hoc networks, but it

is still the aim of the routing algorithm to find the power optimal path. The localized computed Gabriel graph and the localized computed Delaunay triangulation are proved to be power optimal. Planarity is often a required property for position-based routing algorithms. These routing approaches use the planarity of the topology to forward a message through the network. The localized computed Delaunay triangulation is neither planar nor bi-connected, but these properties can be easily achieved. We present in this chapter a distributed algorithm that makes the localized computed Delaunay triangulation planar and bi-connected. We call this topology the *Short delaunay triangulation* $T_{SDT}(S, E)$. The Short delaunay triangulation was first published in [Str05a]. The localized computed Delaunay triangulation as presented in Definition 18 is a supergraph of the Short delaunay triangulation.

Definition 24 (Short delaunay triangulation). *The Short delaunay triangulation (SDT) is a planar and bi-connected subgraph of the localized computed Delaunay triangulation.* $T_{SDT}(S, E) \subseteq T_{DT}(\bar{S}, \bar{E})$, $S = \bar{S}$, $E \subseteq \bar{E}$.

The Short delaunay triangulation fulfills the Delaunay property of Definition 13 locally, but it is not a subgraph of the global Delaunay triangulation $DT(S)$. It contains all edges of $DT(S)$ that are shorter than the communication range and some additional edges which are not present in $DT(S)$. The Short delaunay triangulation has a constant length stretch factor and is power optimal. Each node requires for the computation only information about its single hop neighborhood. $T_{SDT}(S, E)$ is denser than $T_{GG}(S, E)$ and $T_{YG_k}(S, E)$. Hence, the load balance is in this topology better than in the others. The greatest benefit of a topology based on the Delaunay triangulation is the dual graph of the Delaunay triangulation, the Voronoi diagram. The Voronoi diagram can be easily computed from the Delaunay triangulation and is a very appropriate data structure for routing in wireless ad-hoc networks. Details about Voronoi diagrams and routing with Voronoi diagrams can be found in Chapter 7.

6.1 Single-hop Broadcast vs. Point-to-point Communication

In distributed computing, a distinction can be made between two different communication paradigms: the single-hop broadcast model and the point-to-point communication model. A single-hop broadcast is a particular multicast. A multicast forwards a message to a specific set of nodes (Definition 1). In the case of single-hop broadcast this set contains the single hop neighbor nodes. Therefore, in the single-hop broadcast model a node sends a message simultaneously to all nodes

in its neighborhood, whereas in the point-to-point communication model a unicast message is sent over one dedicated communication link to one distinct neighbor node. In our algorithm, we use the point-to-point communication model because of the following two reasons:

- A lot of algorithms broadcast messages to all nodes within the neighborhood. This seems reasonable at the first sight. However, if all of these nodes send an acknowledgment message at the same time the messages interfere. Therefore, at least the Medium Access Control (MAC) protocol divides each broadcast into unicast messages. It seems to us more efficient to avoid broadcasts in wireless ad-hoc networks completely.

- The point-to-point communication model is also advantageous from the viewpoint of security: Wireless networks may be used in hostile environments, where authentication and encryption are necessary to protect the communication against malicious nodes. It is unacceptable in a high-risk environment to reveal the exact location of nodes to anyone within range [CY02]. Therefore, pairwise cryptographic techniques must be employed to protect the position information exchange between any two nodes in position based routing protocols. Several secure routing protocols based on shared keys have been proposed in the literature (e.g., [PH03, KLX+02]), which are much easier to maintain if dedicated communication links can be used.

6.2 Related Work

The distributed algorithm for the computation of the Short delaunay triangulation presented in this chapter was first published in [Str05a].

Li et al. [LCWW03] as well as Gao et al. [GGH+01b] have published two other algorithms for the localized computation of planar and bi-connected topologies with the properties of the Delaunay triangulation. The topologies obtained by these algorithms are very similar to the Short delaunay triangulation and the properties of these topologies are nearly the same. The major difference between these algorithms and our algorithm is that both algorithms use single-hop broadcasts to communicate with the neighbor nodes whereas our algorithm uses dedicated point-to-point communication. Furthermore, we will show in Section 6.4 that the SDT-algorithm is easy upgradeable for the usage in dynamic environments, whereas the algorithms presented in [LCWW03] and [GGH+01b] are not.

The algorithm of Gao et al. [GGH+01b] constructs a planar graph called *Restricted Delaunay Graph (RDG)*. First, each node $s \in S$ computes the local Delaunay triangulation $T_{DT}(s, E_{DT}(s))$ with its single hop neighbors and sends its local Delaunay triangulation to all of its single hop neighbors. Second, each node s deletes an edge (s,t) from $T_{DT}(s, E_{DT}(s))$ if the edge does not exists in the local Delaunay triangulation of a common single hop neighbor. In this algorithm, each node collects the information of all single and double hop neighbors, which produces a lot of overhead in dynamic environments.

Li et al. [LCWW03] propose the *PLDel(S)* as planar subgraph. Again, the nodes compute the local Delaunay triangulations of their single hop neighborhood. Then each node proposes its local Delaunay triangles and waits for reply messages whether the proposed triangles are also valid in the Delaunay triangulations of the neighborhood or not. The main difference appears in the construction idea, however: Li et al. [LCWW03] accept only a subset of edges at the beginning, the Gabriel edges, and then adds the remaining Delaunay edges. Our algorithm starts with all edges and then removes the non-Delaunay edges.

6.3 Algorithm

We describe in this section the distributed algorithm for the computation of the Short delaunay triangulation. First, each node s computes its local Delaunay triangulation with the nodes of its neighborhood $\mathcal{N}(s)$. This step eliminates all edges of the communication graph which are originating at node s and which are definitely not Delaunay edges. The subgraph of local Delaunay triangulation which contains only edges originating at node s is the contribution of node s to the overall topology (Lemma 18). The localized computed Delaunay triangulation $T_{DT}(S,E)$ is neither bi-connected (Lemma 17) nor necessarily planar (Lemma 25). The Short delaunay triangulation algorithm eliminates all edges of $T_{DT}(S,E)$ which are non-Delaunay edges in at least one local Delaunay triangulation. The resulted graph is called Short delaunay triangulation $T_{SDT}(S,E)$, and is bi-connected and planar.

We prove in the following the planarity of $T_{SDT}(S,E)$ and that $T_{SDT}(S,E)$ is bi-connected if $CG(\mathcal{S},\mathcal{E})$ is bi-connected. First of all, we propose a number of Lemmata which are required for the major theorems. The following Lemma 26 shows that a communication edge (s_i, s_j) cannot be part of the Short delaunay triangulation, if it is ignored in just one local Delaunay triangulation.

Lemma 26. *If an edge (s_i, s_j) does not exist in just one local Delaunay triangulation $T_{DT}(S_1, DT(S_1))$, $s_i, s_j \in S_1$, this edge does not exist in any larger*

Delaunay triangulation $T_{DT}(S_2, DT(S_2))$ with $S_1 \subseteq S_2$.

Proof. An edge (s_i, s_j) is not part of the Delaunay triangulation $T_{DT}(S_1, DT(S_1))$ if the "empty circle" property (Definition 14) is violated. This property is also obviously violated in $T_{DT}(S_2, DT(S_2))$, $S_1 \subseteq S_2$. □

It is impossible for a node s to decide on its own if an edge of $T_{DT}(s, E_{DT}(s))$ ($E_{DT}(s)$ contains the edges originating at node s) is a non-Delaunay edge in any other local Delaunay triangulation. However, the following Lemma 27 (from the work of Gao et al. [GGH+01b]) gives us a useful tool to solve this problem.

Lemma 27. *If two edges (s_1, s_3) and (s_2, s_4) cross in $CG(\mathcal{S}, \mathcal{E})$, at least one of the four nodes has communication edges to the three other nodes.*

Proof. Both distances $\|s_1, s_3\|$ and $\|s_2, s_4\|$ are shorter than the communication range R. Assume that they intersect at point p. By the triangle inequality, $\|s_1, p\| + \|s_2, p\| \geq \|s_1, s_2\|$ and $\|s_3, p\| + \|s_4, p\| \geq \|s_3, s_4\|$. Summing these two inequalities, we get

$$\|s_1, s_3\| + \|s_2, s_4\| \geq \|s_1, s_2\| + \|s_3, s_4\|. \tag{6.1}$$

Therefore, $\|s_1, s_2\|$, $\|s_3, s_4\|$, or both have length shorter than the communication range R. Similarly, $\|s_1, p\| + \|s_4, p\| \geq \|s_1, s_4\|$, $\|s_2, p\| + \|s_3, p\| \geq \|s_2, s_3\|$ and by summation

$$\|s_1, s_3\| + \|s_2, s_4\| \geq \|s_1, s_4\| + \|s_2, s_3\|. \tag{6.2}$$

By this inequality, $\|s_1, s_4\|$, $\|s_2, s_3\|$, or both have length shorter than the communication range R. No matter in which case, the node shared by the two short edges has communication edges to all three other nodes. □

The following Definition 25 gives us a useful notation for local Delaunay triangulations:

Definition 25 (correctly computed).

We call a local Delaunay triangulation $T_{DT}(\mathcal{N}(s), DT(\mathcal{N}(s)))$ computed correctly with respect to the final Short delaunay triangulation $T_{SDT}(S, E)$, if

$$T_{DT}(s, E_{DT}(s)) \subseteq T_{SDT}(S, E),\ s \in S.$$

According to Lemma 27, the intersection of communication edges, which must be resolved in the course of planarization, can appear in three different cases:

- **Case 1.** Each node has communication edges to all three other nodes. In this case the local Delaunay triangulation is computed correctly at each node. One of the intersecting communication edges must be a non Delaunay edge,

however. Both corresponding nodes of the non Delaunay edge omit this communication edge in their local Delaunay triangulation. Nevertheless, both corresponding nodes s_i, s_j of the non Delaunay edge (s_i, s_j) generate an "*edge-ignore*" instruction for (s_i, s_j), because the nodes use only local information and do not know whether the corresponding node computed the local Delaunay triangulation correctly or not.

- **Case 2.** Two edges intersect and two nodes have communication edges to the three other nodes. This case makes only sense, if the two nodes with the communication edges to the three other nodes are not connected by one of the intersecting edges. Otherwise no intersection may occur. In Figure 5.4, s_2 and s_3 are such nodes. In this example, the communication edge (s_1, s_3) is a Delaunay edge and edge (s_2, s_4) is a communication edge which is not a Delaunay edge. Node s_2 computes its local Delaunay triangulation $DT(\mathcal{N}(s_2))$ correctly, but node s_4 computes a non-correct Delaunay triangulation $DT(\mathcal{N}(s_4))$. The different local Delaunay triangulations became inconsistent. Only node s_2 generates an "*edge-ignore*" instruction for edge (s_2, s_4) in this case.

- **Case 3.** In the last case only one node has communication edges to all of the three other nodes. This case is a little bit harder to deal with because this node must inform two single hop neighbors that the edge between them must be ignored. Node s_3 is the only node in Figure 6.2 which has communication edges to all other nodes and is therefore the only node which computes its local Delaunay triangulation correctly. The nodes s_2 and s_4 have no communication edges to node s_1, so edge (s_2, s_4) is locally a valid Delaunay edge for both nodes. Node s_3 generates "*edge-ignore*" instructions for both nodes, s_2 and s_4.

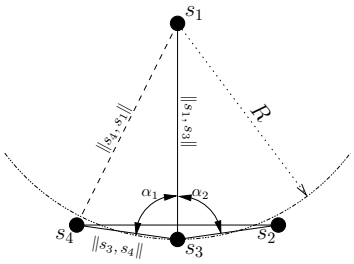


Figure 6.2: *Only node s_3 has communication edges to all other nodes.*

Other cases of two intersecting communication edges cannot occur, because we have assumed that all nodes have the same communication range and that

all each node can communicate with all other nodes within this communication range. A detailed analysis of these requirements is given in Chapter 11.

The pseudocode of the Short delaunay triangulation algorithm appears in Algorithm 1.

Algorithm 1 (Short delaunay triangulation).
code for node s, $s \in S$:

1: *compute* $T_{DT}(\mathcal{N}(s), DT(\mathcal{N}(s)))$
2: *for each* $(s, s_i) \notin T_{DT}(\mathcal{N}(s), DT(\mathcal{N}(s)))$ *do* //*Case 2,* $s_i \in \mathcal{N}(s)$
3: *generate* $ei\_inst(s, s_i)$ //*Case 2*
4: $ei\_msg(s_i) := ei\_msg(s_i) + ei\_inst(s, s_i)$ //*Case 2*
5: *for each* $(s_i, s_j) \notin T_{DT}(\mathcal{N}(s), DT(\mathcal{N}(s)))$ *do* //*Case 3,* $s_i, s_j \in \mathcal{N}(s)$
6: *generate* $ei_inst(s_i, s_j)$ //*Case 3*
7: $ei\_msg(s_i) := ei\_msg(s_i) + ei\_inst(s_i, s_j)$ //*Case 3*
8: $ei\_msg(s_j) := ei\_msg(s_j) + ei\_inst(s_i, s_j)$ //*Case 3*
9: *for each* $ei\_msg(s_i)$ *do* //$s_i \in \mathcal{N}(s)$
10: *send* $ei\_msg(s_i)$ *to* s_i

11: *when* $ei\_msg(s)$ *is received:*
12: *for each* $ei\_inst(s, s_i) \in ei\_msg(s)$ *do* //$s_i \in \mathcal{N}(s)$
13: *omit* (s, s_i) *from* $T_{DT}(s, E_{DT}(s))$

Figure 6.3: The Short delaunay triangulation algorithm.

In the first line of Algorithm 1, each node s computes its local Delaunay triangulation $T_{DT}(\mathcal{N}(s), DT(\mathcal{N}(s)))$ with the nodes of its neighborhood $\mathcal{N}(s)$. This step eliminates all edges of the communication graph $CG(\mathcal{S}, \mathcal{E})$ which are originating at node s and which are definitely not Delaunay edges. Node s either generates an "*edge-ignore*" instructions $ei\_inst(s, s_i)$ when an edge originating at node s does not exist in $T_{DT}(\mathcal{N}(s), DT(\mathcal{N}(s)))$ (Case 2), or when an arbitrary edge $s_i, s_j)$ does not exist in $T_{DT}(\mathcal{N}(s), DT(\mathcal{N}(s)))$ (Case 3). Each node s unifies the "*edge-ignore*" instructions for every neighbor node $s_i \in \mathcal{N}(s)$ to a message $ei\_msg()$ and sends this messages to the corresponding nodes in its neighborhood. Hence, each node sends at most one message to a neighbor node and the message complexity per node of our algorithm is at most $O(n)$. Each node s omits an edge (s, s_i) in its $T_{DT}(s, E_{DT}(s))$, if just one message from a single hop neighbor node contains an "*edge-ignore*" instruction for (s, s_i). The union of all $T_{DT}(s, E_{DT}(s))$, $s \in S$, is the Short delaunay triangulation $T_{SDT}(S, E)$.

The complexity for Case 3 is not optimal, however. Just generating an "*edge-ignore*" instruction when two single hop neighbors in the local Delaunay triangulation have no Delaunay edge between them would unnecessarily increase the time complexity and the message length. Consequently, a test is needed for Case 3 to verify whether a node must really generate "*edge-ignore*" instructions for the other nodes.

Figure 6.4 illustrates an example of Case 3 and shows the critical regions. The critical regions for node s are the shaded sections A and B. Only two nodes, one from section A and one form section B are endangered to have an edge in their local Delaunay triangulations that is not in the SDT. Node s only has to generate "*edge-ignore*" instructions for all edges where one node lies in the first section and the other node lies in the second section.

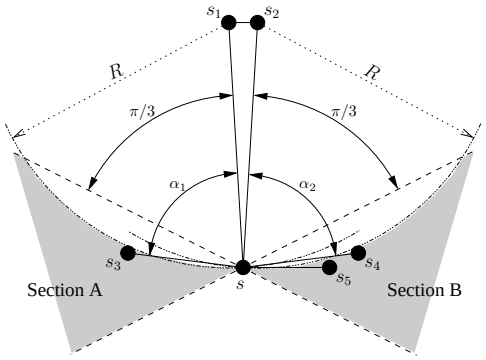


Figure 6.4: *Regions with endangered nodes.*

In the example of Figure 6.4, node s is the only node which has communication edges to two nodes, s_1 and s_2. The number of these nodes is non relevant, it only matters that these nodes are not within the communication range of the nodes in the critical sections (specification of Case 3).

A simple but coarse test to verify whether a node has to generate the "*edge-ignore*" instruction is based upon elementary properties of triangles.

Lemma 28. *If an angle in a triangle is less or equal than $\pi/3$, then the edge opposite to this angle is smaller than or equal to the longest edge in the triangle.*

Proof. The longest side of a triangle is, by the *law of sines*, opposite the largest angle. Since, an angle less or equal than $\pi/3$ cannot be the largest angle in the triangle, the edge opposite to this angle cannot be the longest edge. □

Lemma 28 implies that two single hop neighbors have a common communication edge if the angle between the communication edges to those single hop neighbors is less or equal than $\pi/3$. Therefore, for the existence of Case 3 the following two properties are essential:

- There exists an angle between two successive communication edges (called α_1), which is larger than $\pi/3$.
- There exists a second angle α_2 between two successive communication edges, which is larger than $\pi/3$.

The simple test generates an "*edge-ignore*" instruction for an edge (s_i, s_j), if the angle between s_i and s_j is smaller than π and if it contains α_1 and α_2 as sub-angles.

However, the simple test considers larger regions than the critical regions, i.e., the regions between the dashed lines in Figure 6.4. Node s generates "*edge-ignore*" instruction for (s_3, s_4) and (s_3, s_5).

An additional and more precise test can be used to reduce the number of "*edge-ignore*" instruction further by eliminating the small circle segments between the dashed lines and the critical regions (Figure 6.4).

Consider a network of four nodes that are placed like that in Figure 6.2. If the inequalities of Lemma 29 below are true, then node s_3 is the only node with communication edges to the other nodes and "*edge-ignore*" instructions must be generated.

Lemma 29. *If*

$$\|s_3, s_4\| > 2\|s_1, s_3\| \cos \alpha_1 \tag{6.3}$$

and

$$\|s_2, s_3\| > 2\|s_1, s_3\| \cos \alpha_2 \tag{6.4}$$

then $\|s_1, s_2\| > \|s_1, s_3\|$ *and* $\|s_1, s_4\| > \|s_1, s_3\|$.

Proof. The law of cosines provide us with,

$$\begin{aligned} \|s_1, s_4\|^2 = \|s_1, s_3\|^2 + \|s_3, s_4\|^2 \\ - 2\|s_1, s_3\|\|s_3, s_4\| \cos \alpha_1 \end{aligned} \tag{6.5}$$

where α_1 is the angle between $\|s_1, s_3\|$ and $\|s_3, s_4\|$.

$$\|s_1, s_3\| < \|s_1, s_4\| \tag{6.6}$$

$$\Leftrightarrow \|s_1, s_3\|^2 < \|s_1, s_3\|^2 + \|s_3, s_4\|^2 - 2\|s_1, s_3\|\|s_3, s_4\| \cos\alpha_1 \tag{6.7}$$

$$\Leftrightarrow 2\|s_1, s_3\|\|s_3, s_4\| \cos\alpha_1 < \|s_3, s_4\|^2 \tag{6.8}$$

$$\Leftrightarrow 2\|s_1, s_3\| \cos\alpha_1 < \|s_3, s_4\| \tag{6.9}$$

The proof for node s_2 is the same as for node s_4. □

A node can accurately recognize via Lemma 29 whether two single hop neighbor nodes possibly allow a wrong Delaunay edge.

Therefore, a node s only has to generate an "*edge-ignore*" instruction for a specific edge (s_i, s_j) if the corresponding nodes s_i and s_j lie within the critical sections, if the inequalities of Lemma 29 are true, and if edge (s_i, s_j) is **not** a Delaunay edge in the local Delaunay triangulation $DT(\mathcal{N}(s))$ of node s.

We now show the planarity of the Short delaunay triangulation.

Theorem 1. *The Short delaunay triangulation (SDT) is planar.*

Proof. By Lemma 2, the local Delaunay triangulations are planar. The three cases presented above are the only situations how intersections of edges can occur, according to Lemma 27. If we remove the non Delaunay edges with the algorithm presented above, the resulted graph is planar. □

Corollary 5. $T_{SDT}(S, E) \subseteq T_{DT}(S, E)$

Theorem 1 showed that removing edges form the graph makes the graph planar. We prove in the following Theorem 2 the connectivity of $T_{SDT}(S, E)$.

Theorem 2. *The Short delaunay triangulation is bi-connected.*

Proof. We prove the theorem by showing that the removal of edges from the localized computed Delaunay triangulation, $T_{DT}(S, E)$, done by our algorithm does not disconnect the network. Note that $T_{DT}(S, E)$ is strongly connected if $CG(\mathcal{S}, \mathcal{E})$ is bi-connected (Lemma 17).

The following three cases are the only situations how intersections of edges can occur (Lemma 27).

- **Case 1.** No additional edge is removed from the $T_{DT}(S, E)$. Both corresponding nodes have the edge already removed by the computation of their local Delaunay triangulations.

- **Case 2.** Edge intersections of the second case, only two nodes can communicate with all involved nodes, are the result of inconsistent local Delaunay triangulations, i.e., an edge (s_i, s_j) is present in $T_{DT}(s_i, E_{DT}(s_i))$ but not in $T_{DT}(s_j, E_{DT}(s_j))$. The node that computes its local Delaunay triangulation correctly (assume w.l.o.g. node s_i) sends a message with an "*edge-ignore*" instruction to the corresponding node s_j. The unidirectional edges are removed from the local Delaunay triangulations and the Short delaunay triangulation becomes bi-connected. However, no bidirectional edges become unidirectional, because a bidirectional edge is part of both local Delaunay triangulations and therefore the corresponding nodes do not generate "*edge-ignore*" instructions, and $T_{SDT}(S, E)$ remains connected, since there exists an alternative path in $T_{DT}(S, E)$. (Lemma 17).

- **Case 3.** After the removal of all unidirectional edges (Case 2), all edges in the topology are bidirectional. In the third case, an edge (s_i, s_j) is eliminated from the graph only when there is a third node s_k which sends the "*edge-ignore*" instructions for this edge to both nodes, s_i and s_j. This case eliminates bidirectional edges. An edge is deleted from $T_{DT}(s_i, E_{DT}(s_i))$ and $T_{DT}(s_j, E_{DT}(s_j))$. Hence, unidirectional edges cannot arise. Node s_k has communication edges to s_i and s_j and establishes therefore an alternative path (Figure 6.2). The communication edge to node s_i (resp. node s_j) is either a Delaunay edge or there exists an alternative path to this node in $T_{DT}(\mathcal{N}(s_k), DT(\mathcal{N}(s_k)))$. The Short delaunay triangulation remains connected.

□

6.3.1 Four or more cocircular nodes

The Delaunay triangulation is not unique if four or more sites are cocircular. Each computation algorithm needs a deterministic rule for breaking ties here, i.e., to decide which of the edges are to be included in the Delaunay triangulation and which of them are not. The rule given in Definition 19 is an example of such an deterministic rule.

Four or more cocircular nodes are a problem for our SDT-algorithm, if it is possible that only one node computes its local Delaunay triangulation correctly (Case 3). In such a case, this node must first recognize that four or more nodes

are cocircular, and second, this node must inform the other cocircular node about the non Delaunay edges.

However, we prove in Lemma 30 that only Case 1 and Case 2 can appear, if the nodes of intersecting communication edges are cocircular. At least one node of a crossed edge has communication edges to the nodes of the crossing edge and computes its local Delaunay triangulation correctly. These nodes inform the corresponding nodes of whether the communication edge is a Delaunay edge or not. The computation algorithm of the Short delaunay triangulation therefore needs no additional time and communication. Figure 6.5 gives an example of four cocircular nodes.

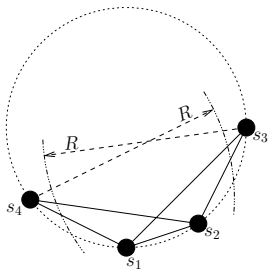


Figure 6.5: *Four cocircular nodes*

Lemma 30. *If two edges intersect and the four nodes are cocircular, than at least two nodes have communication edges to all other cocircular nodes.*

Proof. If two edges (s_1, s_3) and (s_2, s_4) cross, then there is at least one node with communication edges to the three other nodes (Lemma 27). We assume w.l.o.g. that node s_1 is the node with communication edges to s_2, s_3 and s_4. Edge (s_1, s_3) divides the joining circle of the four nodes in two circle segments, whereas node s_2 lies on the perimeter of the first segment and node s_4 lies on the perimeter of the other segment. In at least one segment, the edge (s_1, s_3) is longer than all other chords[1] in this segment. Hence, at least one node of the intersecting edge (s_2, s_4) lies within the communication range of node s_3. If we assume w.l.o.g. that node s_2 is this node, then s_2 has communication edges to s_1, s_3 and s_4. □

[1] A straight line going through two points of a circle is called a secant. Its segment lying inside the circle is called a chord.

6.4 Short delaunay triangulation in Dynamic Environments

An important aspect for the performance of a distributed algorithm is the behavior of the algorithm in dynamic environments. The network topology in wireless ad-hoc networks changes not only via moving nodes, but also by the appearance of new nodes and the disappearance of erroneous nodes. The topology control algorithms presented until today are inefficient in the context of dynamic environments, because these algorithms must restart after each change in the network and rebuild the whole topology (e.g., [LCWW03, GGH+01b]). Successive maintenance of the topology is more efficient and an essential precondition for the usage of topology control algorithms in dynamic environments.

Our topology control algorithm is easy upgradeable for the usage in dynamic environments:

The first step in the SDT-Algorithm (see Algorithm 1) is the computation of the local Delaunay triangulation. This local computation is, just as the computation of the whole topology, more efficient if the algorithm maintains the structure and avoids a continuous recomputation. An overview of various approaches for the dynamic updates of Delaunay triangulation is given in Appendix A.

A change in the network topology generates not necessarily a modification in the Short delaunay triangulation. In many cases, the Delaunay- and non-Delaunay edges in the local neighborhood stay the same and no message must be sent. If a change in neighborhood of node s causes a modification in local Delaunay triangulation $T_{DT}(\mathcal{N}(s), DT(\mathcal{N}(s)))$, node s has only to send the changes in the edge states (from non-Delaunay edge to Delaunay edge and visa versa) to the neighboring nodes and not the whole information about all non-Delaunay edges. Therefore, additionally to the *"edge-ignore"* instruction we need an *"edge-ignore-revoke"* instruction to reactivate a blocked edge. By Lemma 26, a communication edge is not part of the Short delaunay triangulation, if at least one node sends an *"edge-ignore"* instruction for this edge. Hence, a non-Delaunay edge becomes a Delaunay edge if and only if each *"edge-ignore"* instruction for this edge is revoked by the corresponding node or if the transmitter of the *"edge-ignore"* instruction moved out of range. The bandwidth and power costs of our approach are many times smaller than by an approach that distributes the whole topology information at each change of the network.

6.5 Performance Analysis

In this section we analyze the memory, message and time complexity for the computation of the Short delaunay triangulation according to our algorithm. We first analyze the behavior of our algorithm in static environments (no movement and no appearance/disappearance of nodes) and then extend our investigations to dynamic environments.

6.5.1 Message Complexity

- **Static Environment.** At the beginning, each node sends its position to all of its single hop neighbor nodes. The number of nodes in the neighborhood is at most n, so each node sends at most $n-1$ messages[2]. After the computation of $T_{DT}(\mathcal{N}(s), DT(\mathcal{N}(s)))$, each node s generates the required *"edge-ignore"* instructions for the edges in their local neighborhood, unifies the instructions to messages and sends this messages to the corresponding neighbor nodes. To sum up, a node sends at most two messages to a specific neighbor node and the message complexity of a node is therefore $O(n)$.

  Additionally to the number of messages, we must also consider the message length. The total number of nodes in the network is N and we assume a unique node identifier can be represented by $\log N$ bits. Therefore, an *"edge-ignore"* instructions has $O(\log N)$ bits.

  How many *"edge-ignore"* instructions can, in the worst case, be merged to an *"edge-ignore"* message? The number of *"edge-ignore"* instructions is 1 for Case 1 and Case 2. The message length for Case 1 and Case 2 is therefore $C + \log N$ bits[3]. Case 3 (with the critical sections) is a little bit more involved. The longest possible message appears if one node s lies in the first section and the remaining nodes lies in the corresponding section. The single node in the first section receives from node s a message with $C + (n-3) \cdot \log N$ bits. The worst case message length is therefore $O(n \log N)$ bits. However, the remaining nodes in the corresponding critical section receive messages with $C + \log N$ bits (resp. $C + 2 \cdot \log N$ bits if Case 1 or Case 2 also appears). The total number of bits transmitted from node is in this example $O(n \log N)$.

[2] The node itself is also in the neighborhood.
[3] C is some constant.

However, node s has more bits to send, if the other nodes are evenly distributed in two corresponding critical sections. If the number of nodes n is even, each node in a critical section receives a message with $C + (\frac{n-2}{2}) \cdot \log N$ bits (resp. $C + (\frac{n}{2}) \cdot \log N$ bits). If the number of nodes is odd, the nodes in one section receives messages with $C + (\frac{n-1}{2}) \cdot \log N$ bits (resp. $C + (\frac{n+1}{2}) \cdot \log N$ bits) and the nodes in the other section receives messages with $C + (\frac{n-3}{2}) \cdot \log N$ bits (resp. $C + (\frac{n-1}{2}) \cdot \log N$ bits). Hence, the total number of sent bits is in these examples $O(n^2 \log N)$ bits.

- **Dynamic Environment.** After the first computation of the Short delaunay triangulation each node only sends a message to a neighbor node if either the edge to this neighbor node changes its status (Case 1 or Case 2) or an edge originating at this neighbor node changes its status and the conditions for Case 3 are fulfilled. The worst case message length and the total number of sent bits stays also the same, because for one edge a node can either receive an *"edge-ignore"* instruction or an *"edge-ignore-revoke"* instruction. Therefore, the worst case message complexity to adjust the Short delaunay triangulation after a change in the network is $O(n)$ messages, the worst case message length is $O(n \log N)$ bits and the maximal number of bits sent from one node is $O(n^2 \log N)$ bits.

6.5.2 Memory Complexity

- **Static Environment.** Each node only has to store the position information of at most $n - 1$ neighbor nodes and the Delaunay edges. The number of edges in a Delaunay triangulation is linear in the number of sites (Lemma 9). Hence, the memory complexity is $O(n)$.

- **Dynamic Environment.** The memory requirement for the algorithm in the dynamic environment is higher than in the static environment. Again, each node stores the position information of the neighbor nodes. Additionally, the nodes must also store the transmitted and received *"edge-ignore"* instructions. The maximal number of communication edges in a neighborhood of n nodes is $O(n^2)$. Therefore, the memory complexity is $O(n^2)$.

6.5.3 Time Complexity

- **Static Environment.** Each node s computes the local Delaunay triangulation $T_{DT}(\mathcal{N}(s), DT(\mathcal{N}(s)))$ using the nodes of its local Neighborhood $\mathcal{N}(s)$. The number of nodes in the neighborhood is bounded by n and the computational complexity at each node is $O(n \log n)$ (Lemma 4.3).

The SDT algorithm as presented in Algorithm 1 generates an *"edge-ignore"* instructions when two neighbor nodes in the local Delaunay triangulation have no Delaunay edge between them. Hence, we have presented some tests to reduce the number of *"edge-ignore"* instructions. The coarse test uses the principles of triangles (Lemma 28) to verify whether a node lies in a critical section, because critical sections can only occur if two angles between two successive communication edges are greater than $\pi/3$. Hence, each node s computes the $n-1$ angles between the $n-1$ communication edges originating at node s. The more accurate test additionally uses the inequalities of Lemma 29 to verify whether a neighbor node lies within the reduced critical section. However, the number of computations is at most $O(n)$ for each node.

- **Dynamic Environment.** Each node must rebuild the local Delaunay triangulation after each change in the neighborhood. Continual adjustment of the local Delaunay triangulations may be more efficient than rebuilding. An overview of different approaches to maintain the Delaunay triangulation are presented in Appendix A.

Wo kämen wir hin, wenn alle sagten, wo kämen wir hin und niemand ging, um einmal nach zu schauen, wohin man käme, wenn man ginge!

Hans A. Pestalozzi (1929–2004)

Chapter 7

Routing in Wireless ad-hoc Networks

Routing in wireless ad-hoc networks is completely different from routing in wired networks or cellular networks. The network topology in the latter kinds of networks is static, whereas the network topology in wireless ad-hoc networks may change randomly. Hence, routing protocols for wired or cellular networks cannot be used in wireless ad-hoc networks. A wireless ad-hoc network consists of a set of nodes that are dispersed over some area. The nodes communicate with each other over a wireless medium in the absence of a fixed infrastructure and any centralized control. Direct communication between two nodes is only guaranteed if the distance between them is less than their respective communication ranges. It is generally not possible (nor desirable) that all nodes are within the communication range of each other. A wireless ad-hoc network needs a *multi-hop routing* protocol for the communication between non-neighboring nodes. Routing in ad-hoc networks is nontrivial since node mobility and changes in node activity status cause frequent unpredictable topological changes. In the following we list some features for judging the performance of a wireless ad-hoc network routing algorithm:

Loop-freedom. The routing protocols should be inherently loop-free and free from stale routes. The primary goal of every routing algorithm is to deliver the message, and the best assurance one can offer is to design routing algorithms that will guarantee delivery.

Uniform. In a uniform protocol, none of the nodes takes on a distinguished role in the network. No hierarchical structures or clusters are imposed on the network. Uniform protocols are more fault-tolerant than hierarchical protocols, which usually involve the risk of single-point of failure.

Quick access. Each node in the network should have quick access to routes, that is, a long setup time of routes is undesirable.

Localized. Routing should involve a minimum number of nodes, because global state maintenance involves a huge state propagation control overhead and hence is not scalable. Localized routing algorithms are adaptive to changes in the topology caused by mobility or appearance/disappearance of nodes and perform well for wireless ad-hoc networks with arbitrary number of nodes.

Unicast communication. The number of broadcasts made by each node should be limited to a minimum. The obvious and straightforward solution for broadcasts is *flooding*, but flooding yields to the *broadcast storm* problem [NTCS99].

Broadcasts and multicasts cause collisions of acknowledgment messages and are often emulated by unicast messages at the data-link layer.

Low memory requirement. Solutions that require nodes to memorize routes or past traffic are not scalable. It is better to avoid memorizing past traffic at any node, if possible.

7.1 Routing Paradigms

Routing is the topic of wireless ad-hoc networks on which the largest number of publications has appeared. A number of routing protocols are discussed and standardized within the MANET (Mobile Ad hoc NETworks) working group of IETF [MC]. We present in the following a classification of routing protocols and give some well known examples. For a detailed description of the routing protocols we refer to the original papers. An overview of the most common routing algorithms is presented in the work of Royer and Toh [RT99], in the work of Broch et al. [BMJ+98], as well as in the book of Siva Ram Murthy and Manoj [MM04]. However, most routing protocols for wireless ad-hoc networks have been designed for networks of just a few hundreds of nodes and do not scale to networks with thousands of nodes.

Routing protocols for wireless ad-hoc networks can be classified into several types based on different criteria. We use in this work the following classification:

- table-driven vs. on-demand protocols
- link-state vs. distance-vector protocols

Table-driven routing protocols. In a *Table-driven*, or *proactive*, routing protocol, the nodes store routing information for all destinations in the network. Every node maintains the network topology information in the form of routing tables by periodical updates and/or by updates in response of topological changes. Topology information is propagated in the whole network. Whenever a node requires a path to a destination, it runs an appropriate path-finding algorithm on the topology information it maintains. This has the advantage of minimizing delay in obtaining a path to a destination but the updates of topology information consume significant network resources. Moreover, the resources to establish and maintain unused paths are entirely wasted.

On-demand routing protocols. *On-demand*, or *reactive*, routing protocols do not maintain network topology information. These protocols obtain the necessary path when it is required, by using a path establishment process. A node broadcasts a path request and the destination node sends a message back to the requesting node which establishes the path. On-demand protocols do not spend resources maintaining unneeded paths, but the path discovery process brings an unpredictable latency. In order to ensure that the path request reaches the destination, it must be disseminated throughout the network. Flooding the network with messages leads to the well known *broadcast storm problem*. Established paths are maintained by some maintenance procedure to avoid unnecessary flooding.

Link-state routing protocols. The basic concept of *link-state routing* is that every node maintains global connectivity (*link-state*) information of the network. A sender node is aware about a complete path to the destination node.

Distance-vector routing protocols. Instead of connectivity information a node in *distance-vector routing* maintains a distance (hop count or other metric) and a vector (next hop) to a destination. A sender node is not aware of a complete path to the destination. The nodes know only the next hops and forward the message hop-by-hop to the destination.

This classification gives us the following two-by-two matrix.

Category One. Routing protocols of this category maintain, at each node, the complete topology information of the wireless ad-hoc network. Each node propagates its connectivity information to every node in the network either by flooding or by continued information exchange with neighbor nodes. Based on complete topology information in the topology table, any shortest path algorithm can be used to compute the optimal path for each destination.

	table-driven routing	on-demand routing
link-state routing	Category One	Category Two
distance-vector routing	Category Three	Category Four

Table 7.1: Categorization of ad-hoc routing protocols.

Global State Routing (GSR) [CG98] is an example of such a routing protocol. Every node periodically broadcasts its entire topology table to its immediate neighbors. The topology table includes the local connectivity of the node and its current connectivity information for the whole network topology. GSR uses Dijkstra's shortest path algorithm to compute a routing table containing the optimal next hop information for each destination.

Category Two. In routing protocols of this category only the source node is aware of the complete path to the destination node. The source node includes the path in the header of each message and intermediate nodes forward the message according to the path specified in the header.

The *Dynamic Source Routing* (DSR) protocol presented in [JM96] is contained in this category. When a source node has no path to a destination, it broadcasts a route request message. Each intermediate node appends its ID to the message and re-broadcasts it. The destination node sends a reply message, which contains the complete path, to the source node.

Category Three. Routing protocols of this category maintain up-to-date routing information from each node to every other node in the wireless ad-hoc network. Each node is aware of the number of hops and the next hop to all possible destinations within the network.

The *Destination Sequenced Distance-Vector* (DSDV) [PB94] routing protocol is one of the first protocols proposed for wireless ad-hoc networks. Each node periodically broadcasts its current routing table, containing the number of hops and a sequence number for each destination. Each receiving node compares the broadcast sequence number for each destination with the one in its routing table. If the sequence number is higher, the receiver updates its routing table entry, naming the sender as next hop and incrementing the number of hops by one hop. A similar approach is the *Wireless Routing Protocol* (WRP) described in [MGLA96].

Category Four. In routing protocols of this category the source node is only aware of the next hop towards the destination. The source node and the intermediate nodes obtain the routing information by the path establishment process.

A well known routing protocol of this category is the *Ad-hoc On-demand Distance Vector* (AODV) protocol described in [PR99]. When a source node has no path to a destination, it broadcasts a route request message. Nodes receiving the request record a reverse vector back to the node from which the broadcast was received and re-broadcast the request. The destination node sends a reply message to the source node, using the path defined by the reverse vectors. As the reply message follows the reverse path, the corresponding forward vectors are created. Other protocols belonging to this category are *Temporally Ordered Routing Algorithm* (TORA) [PC97] and *Associativity Based Routing* (ABR) [Toh97].

7.2 Position-based Routing

Another routing approach for wireless ad-hoc networks is *geometric* or *position-based* routing. Position-based routing protocols assume the existence of a position service that allows every node to estimate its own coordinates locally, e.g., by GPS. These routing protocols have received significant attention for wireless ad-hoc networking.

For position-based routing protocols a node is not only addressed by the node identifier (ID) but also by the current geographic position of the node. A forwarding node uses the position of the destination given in the packet header as well as the knowledge of its own position and the positions of its single hop neighbor nodes to forward the message towards the destination. Position-based routing requires neither flooding nor the distribution of status information to nodes further than the single hop neighbor nodes. It is not necessary to establish and maintain routes. The nodes neither have to store routing tables nor do they transmit messages to keep the routing tables up-to-date, and no global information about the topology of the network is needed. Forwarding is done “on-the-fly” and only with information about the immediate neighborhood and the position of the destination node. Hence, position-based routing protocols are totally localized and therefore well suited for very large wireless ad-hoc networks. A survey and comparison of position-based approaches can be found in [GSB03, MWH01].

7.2.1 Location Service

However, before routing a message using a position-based routing protocol, the source node needs to retrieve the position of the destination node. Hence, as an essential prerequisite for position-based routing, a *location service* is required from which a source node can acquire the current position of the desired destination node [BCSW98, HL99, LJC+00]. The location service is used to map the unique identifier of a node to its geographical position and distribute this *ID-position* pairs in the network. Such a service has to be scalable to preserve the scalability of position-based routing. A survey on various distributed location services is presented in [MWH01].

Location services can be divided into *flooding-based* and *rendezvous-based* approaches [DPH05]. Flooding-based protocols can be further divided into proactive and reactive approaches. In the proactive flooding-based approach, each (destination) node periodically floods its position to other nodes in the network each of which maintains a position table recording the most recent positions of other nodes. The interval and range of such flooding can be optimized according to the node's mobility and the "distance effect". For example, flooding should be more frequent for nodes with higher mobility, and flooding to faraway nodes can be less frequent than to nearby nodes. The *Distance Routing Effect Algorithm for Mobility* (DREAM) [BCSW98] serves as a good example of proactive flooding-based location services. In reactive flooding-based approaches [KV98], if a node cannot find a recent position of a destination to which it is trying to send a message, it floods a request query in the wireless ad-hoc network in search of the destination.

In rendezvous-based protocols, all nodes (potential senders or receivers) in the network agree upon a mapping that maps the unique identifier of each node to one or more other nodes in the network. The mapped-to nodes are the *location servers* for that node. They will be the rendezvous nodes where periodical position updates will be stored and position queries will be looked up. Two different approaches of performing the mapping, *quorum-based* and *hashing-based*, have been proposed. In the quorum-based approach [HL99], each position update of a node is sent to a subset (update quorum) of available nodes, and a position query for that node is sent to a potentially different subset (query quorum). The two subsets are designed such that their intersection is non-empty, and thus the query will be satisfied by some node in the update quorum. Several methods on how to generate quorum systems have been discussed in [HL99]. In the so-called column-row quorum-based protocol [SV99], the position of each node is propagated in the north-south direction, while any position queries are propagated in the east-west direction. Effectively, the position of each node is disseminated to

$O(\sqrt{N})$ location servers.

In hashing-based protocols, location servers are chosen via a hashing function, either in the node identifier space or in the position space. Hashing-based protocols can be further divided into hierarchical or flat, depending on whether a hierarchy of recursively defined subareas are used. In hierarchical hashing-based protocols, the area in which nodes reside is recursively divided into a hierarchy of smaller and smaller grids. For each node, one or more nodes in each grid at each level of the hierarchy are chosen as its location servers. Position updates and queries traverse up and down the hierarchy. A major benefit of maintaining a hierarchy is that when the source and destination nodes are nearby, the query traversal is limited to the lower levels of the hierarchy. Since the height of the hierarchy is $O(\log N)$, effectively the position of each node is disseminated to $O(\log N)$ location servers. The best example of a hierarchical rendezvous-based location service is the *Grid Location Service* (GLS) [LJC+00].

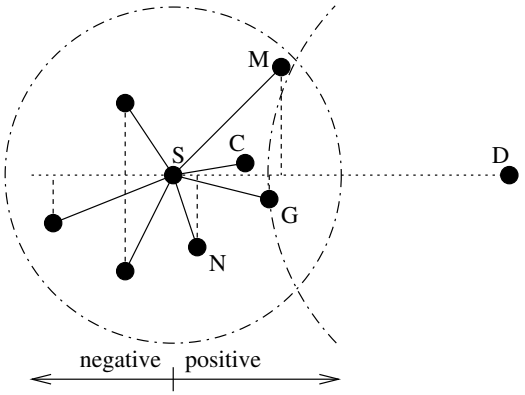
In flat hashing-based protocols (for example, [10], [11], [12]), a well-known hash function is used to map the identifier of each node to a home region consisting of one or more nodes within a fixed position in the network area. All nodes in the home region maintain the position information for the node and can reply to queries for that node; they serve as its location servers. Typically, the number of location servers in the home region is independent of the total number of nodes in the network, and thus effectively the position of each node is disseminated to $O(1)$ location servers. An example of a flat hashing-based protocol is the *Virtual Home Region* (VHR) approach [GH99].

The position information of a location service is more robust than the routes of the routing protocols presented in the previous section. Existing routes can suddenly become unusable because of unpredictable changes in the network topology. The movement or disappearance of only one node can destroy multiple routes. On the other hand, moving or vanishing nodes have low influence on the functionality of the location service. The position of a node changes equably – if at all – and the required position accuracy decreases quickly with increasing distance [BCSW98]. It is not necessary to update the position information of a specific destination node for each forwarding decision.

7.2.2 Greedy Routing

Greedy routing denotes an interesting class of localized, position-based routing algorithms. Each node in a wireless ad-hoc network acquires its own position from its position service, obtains the position of a destination node by a location

service, and receives the positions of its immediate neighbor nodes by simple information exchange. These neighbors are the only nodes with which a node can communicate without using a multi-hop routing approach. Note that the number of nodes in the neighborhood is many times smaller than the overall number of nodes in the network. A wireless ad-hoc network may have thousands of nodes, but the number of neighbors of a specific node may be in the range of ten. The routing decision at a node in the wireless ad-hoc network is only based on its own position, the position of its single hop neighbor nodes and the position of the destination node. Greedy routing does not require the establishment or maintenance of routes: The nodes neither have to store routing tables nor do they transmit messages to keep the routing tables up-to-date, and no global information about the topology of the network is needed. Greedy routing is totally localized and hence scalable. The header of a message contains only the ID-position pairs of source and destination node.



Greedy forwarding	selects neighboring node G
Most Forward within Radius	selects neighboring node M
Nearest Forward Progress	selects neighboring node N
Random Progress Method	selects randomly a node with positive progress
Compass Routing	selects neighboring node C

Figure 7.1: *Greedy Routing Approaches.*

The most intuitive greedy routing protocol is called the *greedy forwarding* strategy, proposed in [Fin87]. With greedy forwarding, a node selects as next hop the single hop neighbor that is closest to the destination node among all its neighbors, i.e., the neighbor node with the shortest euclidean distance to the destination node.

Other approaches of greedy routing use the *progress* of a node instead of the euclidean distance. Given a forwarding node S and a destination node D, the progress of node M is defined as the projection onto the line connecting S and D. In the *Most Forward within Radius* (MFR) scheme [TK84], the message is forwarded to the neighbor whose progress is maximal. Hou and Li [HL86] proposed the *Nearest Forward Progress* (NFP) routing protocol, in which a message is sent to the node with the smallest positive progress. In the *Random Progress Method* proposed by Nelson and Kleinrock [NK84] are messages routed with equal probability to a single hop neighbor node with positive progress. The progress routing protocols as well as the greedy forwarding strategy presented above are loop-free.

The *Compass Routing* protocol [KSU99] uses the *direction* of the neighbor nodes to forward the messages. A forwarding node selects the neighbor, such that the direction to the neighbor is closest to the straight line between forwarding node and destination. However, protocols that forward the message to the neighbor with the closest direction are not necessarily loop-free [KSU99].

When successful, greedy forwarding and MFR choose paths which are similar to the optimal path between source and destination founded by a shortest path algorithm. If greedy routing successfully delivers a message to the destination, the required number of steps is $O(N)$. N is the total number of nodes in the wireless ad-hoc network. All proposed greedy routing methods have high delivery rates for dense graphs, i.e., if the spatial density of network nodes is high in relation to the transmission range, and low delivery rates for sparse graphs.

Greedy routing can be used until the message reaches the destination or a *local minimum* with respect to the distance to the destination node. At a local minimum no neighbor is closer to the destination than the node itself and greedy routing is no longer possible. To counter this problem it has been suggested that the message should be forwarded to the node with the least negative progress [TK84] if no nodes with positive progress can be found. However, this raises the problem of looping messages which cannot occur when messages are forwarded only to nodes with positive progress. In the *Geographic Distance Routing* protocol (GEDIR) proposed in [SL01], the message is dropped if the best choice for a forwarding node is to return the message to the node the message came from. Other researchers proposed not to forward messages which have reached a local minimum at all [HL86]. Obviously, none of these proposals can guarantee the delivery of messages. A reliable routing protocol needs a recovery strategy to escape from local minima and to forward the massage to the destination node.

The *Geographical Routing Algorithm* (GRA) proposed by Jain, Puri and Sengupta [JPS01] stores each local minimum node paths to every node it cannot reach with greedy forwarding. The authors propose two route discovery strategies: *breadth-first search* and *depth-first search*. A local minimum node floods with the breadth-first search strategy the network to obtain a path to the destination node. This strategy is equivalent to the path discovery process of reactive routing protocols (see Section 7.1). With the depth-first search strategy each node adds its ID to the discovery message and forwards the message to a neighbor who has not seen the message before. If a node has no possibilities to forward the message, it removes its ID from the packet and returns the message to the node from which it originally received it. If a node receives a same packet twice, it refuses it. Eventually, this strategy yields to a single acyclic path from the local minimum node to the destination node.

Another recovery strategy is the *Intermediate Node Forwarding (INF)* mechanism [CM01]. If a node cannot forward a message to a node closer to the destination than itself, the local minimum picks an intermediate point at random and forwards the message to this point using a greedy forwarding strategy. When the message reaches this intermediate point, a new attempt is started to send the message to the destination. The *Terminode Routing* approach presented in [BGB02] is similar to the INF mechanism, but uses anchor points instead of random intermediate points. Each node keeps a list of other nodes (friends) to which it maintains a path. These friends are used by a path discovery protocol to find an anchored path from a source node to a destination node. If the anchors are set correctly, a message follows an anchored path (from one anchor to another) until it eventually reaches the destination node. Greedy forwarding is used to deliver a message from one anchor point to another. The Terminode Routing approach further needs a path maintenance method that allows a node to improve acquired paths and delete obsolete or malfunctioning paths.

7.2.3 Perimeter Routing

Perimeter routing, also known as *Face routing* or *Compass routing*, requires a planar topology to forward the message. Algorithms for the computation of a planar topology are presented in Chapter 5 and Chapter 6. A connected planar topology partitions the plane into faces that are bounded by polygons. The perimeter routing approach uses the Right hand rule to forward a message along the perimeter of the faces that are crossed by the (imaginary) straight line connecting the source node S and the destination node D. Each node that receives a message forwards the message on the next edge counterclockwise with respect to itself from the ingress edge. The Right hand rule traverses the interior of a closed polygonal face

in clockwise edge order or in counterclockwise edge order if it is the exterior face.

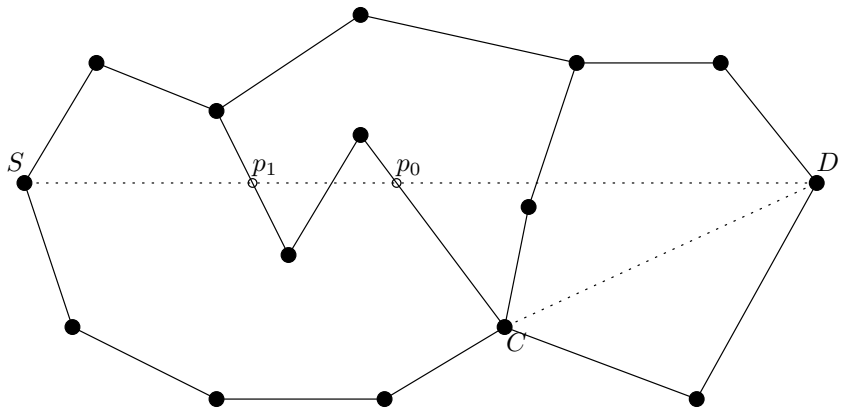


Figure 7.2: *Perimeter Routing Approaches.*

The destination node is not often part of the current face. Hence, the message must be brought to a face closer to the destination. Different variants of perimeter routing exist to decrease the distance to the destination. The point of the face switch is stored in the message header. The following enumeration presents three different approaches proposed in literature:

A. In the *Compass Routing II* approach [KSU99], the message traverses the entire face to determine the closest intersection point p_0 of the face boundary with line $\overline{SD}$. The message returns to the first node of the edge containing p_0 after the exploration of the current face and the node associated to this edge forwards the message along the other face adjacent to this edge.

B. Instead of the closest intersection point, the *Other Face Routing* approach [KWZ03b] explores the face for the closest node C to the destination node. After completing of the exploration of the entire face, the message returns to node C and is forwarded along the perimeter of the face that is crossed by line $\overline{CD}$.

C. The *Face-2* approach presented by Bose et al. [BMSU01] traverses the face until it reaches the edge with the first intersection point of the face boundary and line $\overline{SD}$ (p_1 in Figure 7.2). The node associated to this edge forwards the message along the perimeter of the other face adjacent to the edge containing p_1.

Perimeter routing guarantees message delivery if the topology is planar and connected. All three approaches are totally localized and hence scalable. The first

and the second approach completely traverse a face at least one time, whereas approach C does not traverse a face completely. However, in unfavorable topologies the performance of the third approach can be worse than the performance of approach A and B. First of all, we show in the following lemma that perimeter routing approach A and approach B terminate in $O(N)$ steps.

Lemma 31. *(Kranakis et al. [KSU99]) Perimeter routing approach A and B terminates on a planar graph in $O(N)$ steps. N is the total number of nodes in the wireless ad-hoc network.*

Proof. A face is left at that intersection point of the face boundary with the straight line connecting source and destination that is closest to the destination node in the first approach. In approach B, a face is left at the closest node of the face to the destination node. In both approaches, the next face contains always at least one intersection point or node closer to the destination. Therefore, no face is visited twice. A planar graph has at most $3N - 6$ edges (Euler's Theorem) and each edge is in at most 2 faces. Therefore, each edge is visited at most 4 times. The approaches terminate in $O(N)$ steps. □

In the third approach, the faces are switched at the first intersection point of the face boundary with the straight line connecting source and destination. This approach also terminates in a finite number of steps, since the distance to the destination is decreasing during each switch, but in unfavorable topologies the performance can be worse than in the other two approaches.

Lemma 32. *(Bose et al. [BMSU01]) Perimeter routing approach C terminates in inauspicious topologies in $\Omega(N^2)$ steps.*

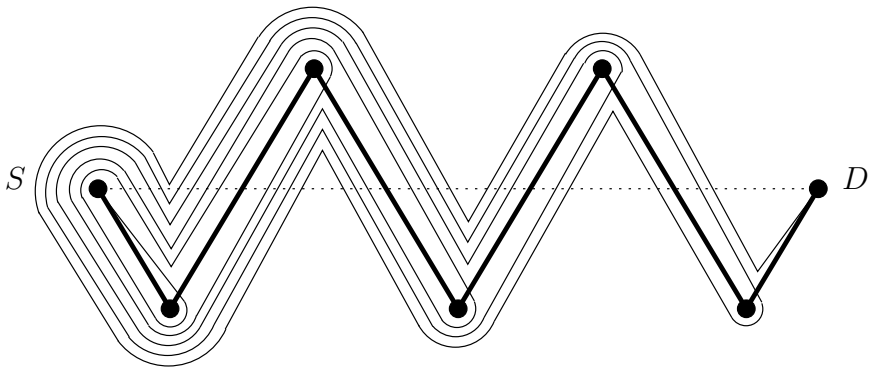


Figure 7.3: *Unfavorable topology for perimeter routing approach C.*

Proof. An inauspicious topology is shown in Figure 7.3. The thin line represents the path of a message forwarded according to perimeter approach C. The path from S to D crosses the (imaginary) straight line $\overline{SD}$ many times. The message returns at each intersection point to node S and reruns the whole path. In a topology like Figure 7.3 with N nodes requires a message $\frac{N*(N+1)}{2}$ steps to be forwarded from S to D . □

The chosen paths in all approaches can be very long. The *Adaptive Face Routing* (AFR) protocol presented by Kuhn et al. [KWZ02] uses the boundary of an ellipse with source node and destination node as foci to restrict the path to edges inside the ellipse. The size of the ellipse is increased if no path is found.

The planarity of the topology is an indispensable property to guarantee the reliability of perimeter routing. A comprehensive analysis of this fact is given in Section 11.

7.2.4 Greedy/perimeter Routing

Karp and Kung [KK00] propose the *Greedy Perimeter Stateless Routing (GPSR)* protocol. GPSR uses the above mentioned greedy forwarding strategy until the algorithm reaches a local minimum where no neighbor is closer to the destination than itself. At such a local minimum, greedy routing fails and GPSR uses a perimeter routing approach as recovery strategy. If perimeter routing reaches a node whose position is closer to the destination node than the node where the greedy strategy previously failed, the greedy routing algorithm takes over control again. At this node, the message can continue greedy forwarding towards the destination without the danger of returning to the prior local minimum. The idea of combining greedy and perimeter routing on planar graphs is independently investigated by Bose et al. [BMSU01]. Both use a localized computed version of the Gabriel graph [GS69] as underlying graph. Karp and Kung also proposed the Relative neighborhood graph [Tou80] as alternative planar graph. The *Greedy Other Adaptive Face Routing* (GOFAR) algorithm [KWZ03b] combines the greedy forwarding strategy with the AFR protocol.

The header of a message contains additional information such as the position of the node where it entered the perimeter mode and the position of the last intersection that caused a face change. Scenarios like in Figure 7.3 cannot occur in a greedy/perimeter routing protocol, because perimeter routing is only the recovery strategy. Greedy/perimeter routing forwards the message in such scenarios in greedy mode.

Es gibt nichts Praktischeres als
eine gute Theorie.

Immanuel Kant (1724 - 1804)

Chapter 8

The Voronoi-Aided Routing Protocol (VAR)

This chapter gives an efficient implementation of the greedy/perimeter routing approach presented in Section 7.2. The novel implementation is called Voronoi-aided routing (VAR) protocol and was published in [Str04b] and [Str04a] for the first time. The Voronoi-aided routing protocol uses the Voronoi diagram and the dual of the Voronoi diagram, the Delaunay triangulation, to improve the performance and the efficiency of greedy/perimeter routing.

8.1 The Voronoi diagram, Post offices, and Duality

8.1.1 Voronoi diagram

The Voronoi diagram belongs to classical problems of computational geometry. The Voronoi diagram is named after the mathematician M. G. Voronoi who explored this geometric construction in 1907 [Vor07, Vor08, Vor09]. However, as early as in 1850 another mathematician, G. L. Dirichlet, studied the problem. Accordingly the Voronoi diagram is sometimes named Dirichlet tessellation. The history of Voronoi diagrams, their properties, algorithms for their construction, as well as a variety of different applications for computer science and other scientific areas are proposed in the book of Edelsbrunner [Ede87], in the survey of Aurenhammer [Aur91] and in many other publications about computational geometry.

Definition 26 (Voronoi diagrams). *Let S be a set of N distinct sites. The Voronoi diagram of S, denoted $VOR(S)$ is the subdivision of the plane into n Voronoi cells, one for each site $s_i \in S$. A point p lies in the cell corresponding to a site $s_i \in S$, if $\|p, s_i\| < \|p, s_j\|$, for each $s_i, s_j \in S, j \neq i$. Herein, $\|p, q\|$ is the Euclidean distance from node p to node q.*

Properties of Voronoi diagrams:

- **Voronoi edge:** A point p lies on a Voronoi edge between sites s_i and s_j if and only if the largest empty circle centered at p touches only s_i and s_j. A Voronoi edge is a subset of locus of points equidistant from s_i and s_j.
- **Voronoi vertex:** A point p is a Voronoi vertex if and only if the largest empty circle centered at p touches at least 3 sites. A Voronoi vertex is an intersection of 3 or more Voronoi edges, each equidistant from a pair of sites.
- **Voronoi cell:** A Voronoi cell is a (possibly unbounded) convex polygon. The boundary of a Voronoi cell consists of Voronoi edges and Voronoi vertexs.

The Voronoi diagram is among the most important geometrical structures in two-dimensional computational geometry, and many construction algorithms have been proposed in the last twenty years. The worst-case complexity of computing a Voronoi diagram for N sites in two dimensions is $O(N \log N)$ and the storage requirement is only $O(N)$ [Aur91]. An optimal method for the computation of Voronoi diagrams is *Fortune's plane sweep* algorithm [For87].

8.1.2 Point location

Point location has been referred to as the *post-office problem* in Knuth [Knu97], Volume One. Given is a set S of n sites in the plane (post offices), the problem is to find the site which is closest to a given query point p (the location of a person). Note that there exists a trivial $O(n)$-time solution by computing all n distances. Point location is one of the best known problems in the context of Voronoi diagrams. A Voronoi diagram divides the plane in different convex polygons. A point p is closest to a site s_i, $s_i \in S$ if and only if p falls into the Voronoi cell of s_i. Hence, to find the site closest to p, it suffices to determine the Voronoi cell that contains p. There exist worst-case optimal techniques to perform point location on a Voronoi diagram, such as the *triangulation refinement method* due to Kirkpatrick [Kir83], and the *bridged chain method* due to Edelsbrunner et al. [EGS86].

For greedy routing, a node uses its own position and the positions of its neighbor nodes to compute a local Voronoi diagram. When a node wants to forward a message to a destination node, it determines the cell the destination node belongs to (using the point-location algorithm), and forwards the message to that unique neighbor node that is the site of this cell. If two or more neighbor nodes are

equidistant from the destination node, one of the neighbor nodes is either chosen arbitrarily or by some deterministic rule (e.g. node ID).

Point location in a Voronoi diagram with n regions is possible in $O(\log n)$ time and needs $O(n)$ storage. The post-office problem can hence be solved in logarithmic query time and without increasing the $O(n)$ storage requirement of the construction algorithm. This is asymptotically optimal since it matches the information-theoretical lower bound.

Using any of the optimal Voronoi diagram algorithms presented above we obtain the following result related to the point location problem.

Lemma 33. *Given a set S of n sites in the plane, one can, within $O(n \log n)$ time and linear storage $O(n)$, construct a data structure that supports nearest neighbor queries: For an arbitrary query point p, its nearest neighbor in S can be found in time $O(\log n)$.*

8.1.3 Duality Voronoi diagram and Delaunay triangulation

The Delaunay triangulation is the dual of the Voronoi diagram. For a set of sites in the plane, the Voronoi diagram partitions the plane into convex polygon cells and each cell contains all points closer to the site of this cell than to any other site. Voronoi was also the first to consider the *dual* of this structure, where any two sites are connected whose Voronoi cells have an edge in common. Later, Delaunay obtained the same by defining that two sites are connected if and only if they lie on a circle whose interior[1] does not contain any site of the set (Definition 13). After him, the dual of the Voronoi diagram has been denoted *Delaunay Tessellation* or Delaunay triangulation. Figure 8.1 shows an example of a Delaunay triangulation with the respective Voronoi diagram.

For various reasons, it is much more convenient to express construction algorithms in terms of the Delaunay triangulation than the Voronoi diagram. One reason is to avoid manipulating computed values: because of arithmetic inaccuracies in computations, Voronoi vertexs may be poorly determined if the defining sites are close to being affinely dependent[2]. Another reason is that it is simpler to manipulate a cell complex with regular bounded cells than one with irregular unbounded cells. The Voronoi diagram can be obtained in linear time from the Delaunay triangulation. Some data structures (e.g., the *Quad-Edge* structure of

[1] We consider that the interior of a circle is an open disc, i.e., the boundary is excluded.

[2] Coordinates are called affine dependent, if it is possible to represent one coordinate as linear combination of other coordinates. E.g., three collinear vertices.

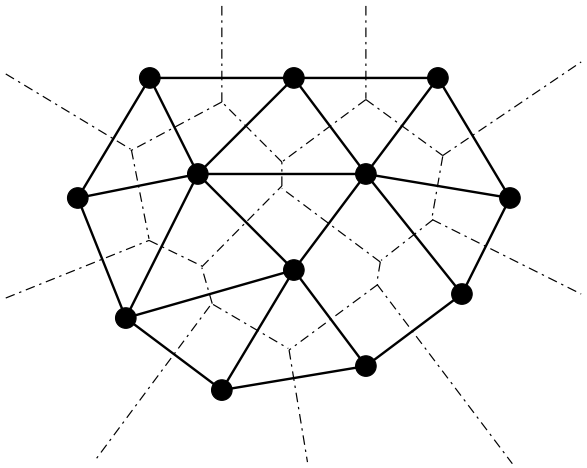


Figure 8.1: *Delaunay triangulation and Voronoi diagram*

Guibas and Stolfi [GS85]) stores simultaneous the Delaunay triangulation and the Voronoi diagram.

8.2 The SDT/VAR Communication Protocol

Two of the common problems in the context of wireless ad-hoc networks are *topology control* and *routing*. The aim of topology control (Chapter 3) is the construction of an appropriate topology to improve the efficiency and performance of the routing algorithm and routing manages the message exchange between non-neighboring nodes in a wireless ad-hoc network (Chapter 7).

The Voronoi-aided routing (VAR) protocol is an efficient implementation of the greedy/perimeter routing approach. It uses Voronoi diagrams to improve the efficiency of greedy forwarding and requires a planar topology for perimeter routing.

Algorithms proposed until today are either algorithms for topology control or algorithms for routing. Our work is the first approach that combines topology control and routing. The computation of the Short delaunay triangulation (SDT) automatically yields to local Voronoi diagrams for the efficient implementation of greedy routing. A common implementation based upon one efficient data structure and the VAR-protocol presented in this section leads to considerably increased performance.

Voronoi-aided routing is a position-based routing algorithm which reliably delivers a message from the source node to the destination node without any routing information at intermediate nodes. The VAR protocol is a totally localized routing algorithm: A node makes forwarding decisions only based on its own position, the positions of its neighbors and the position of the destination. VAR does not maintain any information except the Short delaunay triangulation and is very efficient in terms of computing time and memory. If there are no partitions in the network, each message will eventually reach its destination.

SDT/VAR is a very efficient implementation of the greedy/perimeter routing approach. It uses the Short delaunay triangulation algorithm presented in Chapter 6 for the computation of its local contribution to the overall topology. A node requires for the computation the positions of its single hop neighbor nodes in the communication graph. The number of single hop neighbor nodes is, according to Definition 5, bounded by n and n can be many times smaller than the total number of nodes in the network. Hence, the computation of the SDT has time complexity $O(n \log n)$. This is the lower bound on the construction of a planar topology. The dual graph of the local contribution is called the local Voronoi diagram. The local Voronoi diagram of a node has only the node itself and the single hop neighbor nodes in the topology as sites. The number of neighbor nodes in the topology, we denote it with n', can be significantly lower than n. In the Delaunay triangulation, n' is on average 6 (Lemma 10).

8.2.1 Greedy forwarding with Voronoi diagrams

In greedy forwarding, see Section 7.2.2, a node forwards a message to the neighbor node that is closest to the destination node, i.e. the neighbor node with the shortest Euclidean distance to the destination node among all its neighbor nodes. The question of how to find this neighbor node is usually not addressed, however. In our approach, a node requires only its local Voronoi diagram and a point location algorithm (as defined in Section 8.1.2) to acquire the subsequent node in greedy mode. The local Voronoi diagram is a priori given by the computation of the topology and the point location algorithm needs only $O(\log n')$ time to find the next node in the communication path.

The pseudocode of greedy forwarding appears in Algorithm 2. The header of a message M contains the destination node $dest(M)$ of the message and function $PointLoc(p)$ determines the closest node to point p.

Each node uses only its local Voronoi diagram to forward a message. The lo-

Algorithm 2 (Voronoi-aided routing (greedy forwarding)).
code for node s, $s \in S$:

1: $\hat{s} := PointLoc(dest(M))$
2: *send* M *to* $\hat{s}$

Figure 8.2: The greedy routing algorithm.

cal Voronoi diagram divides the plane into Voronoi cells and each cell contains all points closer to the site of this cell than to all other sites. When a node s wants to forward a message M to a destination node $dest(M)$, either received from node $\breve{s}$ or new generated ($\breve{s} := s$), it determines the cell the destination node belongs to (using $PointLoc()$), and forwards the message to the unique neighbor node $\hat{s}$ this cell belongs to. If two or more neighbor nodes are equidistant from the destination node, i.e., if the destination node lies on a Voronoi edge, one of the neighbor nodes is either chosen arbitrarily or by some deterministic rule (e.g. value of node ID). It is obvious that forwarding with this strategy is only possible if the message does not reach a local minimum (with respect to the distance to the destination node).

In Lemma 34 below, we will show that the greedy routing approach delivers a message reliably to the destination if the network is dense.

Definition 27 (Dense Network). *A network is called dense if each node s in the network has, for each destination node (that is not a single hop neighbor node), a single hop neighbor which is closer to the destination than node s itself.*

Lemma 34. *In a dense network, the greedy forwarding algorithm eventually delivers each message to its destination.*

Proof. The greedy forwarding algorithm is loop-free in dense networks, because each forwarding step decreases the distance to the destination node: If a node would receive the same message twice, at least one forwarding step had increased the distance to the destination node, which is a contradiction. Hence, each message is eventually delivered to its destination. □

Greedy forwarding can be used until the message eventually reaches the destination node. Obvious, greedy forwarding terminates in a dense network in $O(N)$ steps, because messages are never sent backwards. N is the total number of nodes in the wireless ad-hoc network. Unfortunately, it may happen in arbitrary networks that the best choice for forwarding is the current node itself, i.e., that the message is caught in a local minimum. The greedy forwarding routing protocol

requires a recovery strategy to escape from the local minimum.

In contrast to other routing algorithms the usage of Voronoi diagrams gives us a handy specification of a local minimum:

Definition 28 (local minimum). *A node s is a local minimum according to a specific destination, if the destination lies in the Voronoi cell of node s.*

8.2.2 Perimeter Routing using the Short delaunay triangulation

Greedy forwarding will fail at a local minimum, i.e., when no neighbor is closer to the destination node than the node itself. The message has to travel backwards in order to escape from this void region in the network topology. A good routing algorithm needs a reliable recovery strategy to deliver a message to the destination node.

We use the Short delaunay triangulation (SDT), a topology based on the construction rules of the Delaunay triangulation and presented in Chapter 6, as underlying planar topology. The Delaunay triangulation is the densest proximity graph and the faces are triangles. In the global Delaunay triangulation, a local minimum as defined in Definition 28 cannot occur, because the Voronoi cell of a node does not contain any other node. Only in localized computed topologies like the SDT, where edges are restricted by the communication range, a node could be a local minimum. Voronoi-aided routing (VAR) uses perimeter routing to escape from the local minimum and to find a node closer to the destination than the node where the greedy forwarding previously failed. At this node, the message is once again forwarded in greedy mode. Section 7.2.3 presents three different approaches for perimeter routing. The recovery strategy of the VAR algorithm is perimeter routing approach C, presented by Bose et al. [BMSU01] as *Face-2* approach.

However, it does not matter which approach is used in the VAR protocol, because face changes are quite rare. The need for face changes in greedy/perimeter routing will be discussed in Lemma 36. The VAR protocol only uses perimeter routing as recovery strategy, the perimeter mode is ended as soon as a closer node to the destination node is reached. Greedy forwarding restarts at the first node closer to the destination, but the presented perimeter routing approaches can be used to find a “better” node to revert to greedy forwarding. The perimeter approaches presented in Section 7.2.3 offer solutions to find the first intersection point of the bordering face and the line connecting start node of perimeter routing and destination node, or the closest intersection point of the bounding face and

this line, or to find the closest node on the face. Each approach can improve the performance of the simple VAR protocol. However, those approaches have also a big drawback: Nodes in wireless ad-hoc networks are not uniformly distributed, so it is possible that large regions are totally empty of nodes. The enclosing face of such an empty region can contain a huge number of nodes. In some wireless ad-hoc network this face can be the exterior face of the network. In networks with thousands of nodes, perimeter routing approaches that initially explore large parts or the entire face can be very inefficient.

The pseudocode of greedy/perimeter routing appears in Algorithm 3. The header of a message contains the following fields:

$dest(M)$	destination node of M
$mode(M)$	forwarding mode, either $greedy$ or $perimeter$
$locMIN(M)$	node where message entered in perimeter mode
$cross(M)$	position of the last intersection that caused a face change

Function $RHR()$ determines the next node $\hat{s}$ in perimeter routing. If a node s receives a message from node $\breve{s}$ in perimeter mode, $RHR(s, \breve{s})$ returns the associated node from the next edge counterclockwise with respect to the ingress edge. If perimeter mode starts at node s, $RHR(\overline{s\, dest(M)})$ returns the associated node from the next edge counterclockwise with respect to the imaginary line $\overline{s\, dest(M)}$.

If greedy forwarding reaches a local minimum at node s, the position of node s is stored in the field $locMIN(M)$ of the message header and M is forwarded along the perimeter of the face that is crossed by the (imaginary) straight line connecting node s and the destination node $dest(M)$. Node s also stores its position in $cross(M)$, because s is the first intersection point with line $\overline{locMin(M)\, dest(M)}$. The message is forwarded in perimeter mode until a node is reached whose position is closer to the destination than the position of the local minimum (stored in $locMIN(M)$). At this node, the message can continue greedy forwarding towards the destination without danger of returning to the prior local minimum. According to the routing principles of approach C (Section 7.2.3), no message is forwarded along an edge crossing the (imaginary) line $\overline{locMIN(M)\, dest(M)}$. While forwarding around the face, each node determines whether the edge to the chosen next node $\hat{s}$ intersects $\overline{cross(M)\, dest(M)}$. If node s is adjacent to an edge intersecting $\overline{cross(M)\, dest(M)}$, s stores the new intersection point in $cross(M)$ and tries to forward the message along the next face bordering the intersected edge. Each node can make all routing decisions based only on the information about its local neighbors.

Algorithm 3 (Voronoi-aided routing).
code for node s, $s \in S$:

```
1:  if s ≠ dest(M) than
2:    if mode(M) == greedy then
3:      ŝ := PointLoc(dest(M))
4:      if s == ŝ then                          // s is a local minimum
5:        mode(M) := perimeter
6:        locMIN(M) := s
6:        cross(M) := s
7:        ŝ := RHR(cross(M) dest(M)‾)
8:    else
9:      if ‖s, dest(M)‖ ≥ ‖locMIN(M), dest(M)‖ then
10:       mode(M) := greedy
11:       goto 2:
12:     else
13:       ŝ := RHR(s, s̆)
14:       while (s, ŝ) crosses cross(M) dest(M)‾ do
15:         cross(M) := intersection point (s, ŝ), locMIN(M) dest(M)‾
16:         ŝ := RHR(s, ŝ)
17:   send M to ŝ
```

Figure 8.3: The greedy/perimeter routing algorithm.

We prove in the following that the perimeter algorithm always finds a node closer to the destination node if the network has no partitions.

Lemma 35. *A message forwarded with perimeter routing eventually reaches a node closer to the destination than the node starting perimeter mode, as long as the topology is planar and connected.*

Proof. The proof is by contradiction. Consider a face, where no node is closer to the destination than the starting node of the perimeter mode. In such a face the message returns to the starting node.

We have to distinguish two different cases. The two cases are outlined in Figure 8.4. Note that, in general, a face can contain many more nodes than the faces in Figure 8.4, but the conclusion is still the same. Node s_0 is a local minimum for the destination node (s_D). The routing mode is changed to the perimeter mode and node s_0 forwards the message with the Right hand rule. In the first case (Figure 8.4; dashed lines), the message is forwarded to node s_1, hand over to s_2

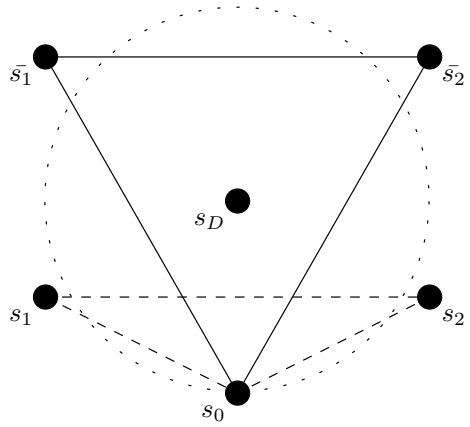


Figure 8.4: *In this two cases the Right hand rule generates a circle.*

and finally back to node s_0 – a circle. But this case cannot happen, because edge (s_1, s_2) crosses the imaginary line connecting node s_0 and the destination node s_D and the message, according to the perimeter routing approaches presented in Section 7.2.3, will be forwarded along the perimeter of the adjacent face bordering edge (s_1, s_2). In the second case (Figure 8.4; solid lines) the destination node is inside the face. The message circulates along the perimeter of the enclosing face, because no node is closer to the destination than the starting node of the perimeter mode. In this case, either the destination node is partitioned from the encircling nodes, or the destination node is connected to the network via a node beyond the encircling face. This is either a contradiction to the postulated connectivity or a contradiction to the postulated planarity. □

The necessity of face changes depends primary on the underlying topology. Face changes in greedy/perimeter routing are only required if the underlying topology is based on the Delaunay triangulation, like the Short delaunay triangulation. In all other presented planar topologies, the Euclidean minimum spanning tree, the Relative neighborhood graph, and the Gabriel graph, face changes are unnecessary, as the following Lemma 36 shows:

Lemma 36. *A face change cannot occur in the localized computed topologies:*

$$T_{GG}(S, E), T_{RNG}(S, E), \textit{and } T_{EMST}(S, E).$$

Proof. Assume $T_{GG}(S, E)$ as underlying topology, a local minimum s_0 for the destination node s_D, and an edge (s_1, s_2) that crosses the imaginary line connecting node s_0 and the destination node s_D (The assumptions are similar to Figure 8.4). Node s_0 forwards the message in perimeter mode to node s_1. If

s_1 is closer to s_D than node s_0, the message will be forwarded in greedy mode. Hence, a face change can only occur if s_1 is not closer to s_D than s_0. Node s_2 can either be closer to the destination node than node s_0 or not. In the first case $\|s_0, s_2\| \leq \|s_1, s_2\|$. Hence, s_0 must be able to communicate with node s_2 and cannot be a local minimum. In the second case are both nodes, s_1 and s_2, farther away from s_D than s_0. Hence, edge (s_1, s_2) either does not cross line $\overline{s_0\ s_D}$, or node s_0 lies within the critical section of edge (s_1, s_2) and cannot be part of $T_{GG}(S, E)$.

$T_{RNG}(S, E)$ and $T_{EMST}(S, E)$ are subgraphs of $T_{GG}(S, E)$. Hence, edge (s_1, s_2) cannot be part of these topologies. □

The necessity to face changes is only required for topologies based on the Delaunay triangulation, because only in these topologies both nodes of a crossing edge could be farther away from the destination node than the local minimum. This fact is discussed in Section 11.3.2.

This chapter completes the first part of the thesis. We have proposed the principles of position-based communication, have presented the Short delaunay triangulation and an efficient implementation of the greedy/perimeter routing approach (VAR algorithm). Next we will examine how faulty nodes and imprecise position information can affect the behavior of a position-based communication approach.

Einen Fehler machen und ihn nicht korrigieren, erst das heißt einen Fehler machen.

Konfuzius (551–479 v.Chr.)

Chapter 9

Fault-tolerant greedy/perimeter Routing

Most existing wireless ad-hoc network routing protocols assume that communication links and nodes operate correctly during message delivery. However, such assumptions usually do not hold in realistic environments. Communication links in wireless networks are unreliable and nodes can become faulty. The loss rate on wireless communication links is much higher than that of wired links, and this effect accumulates quickly as the number of hops increases. For example, when loss rate is 10% per hop, after 15 hops loss rate becomes 80%! Hence, a fault-tolerant routing approach becomes essential especially in large networks of several hundred nodes. Otherwise, the benefits of localized topology control and routing are lost.

9.1 Retransmission

Retransmission is a well known technique to increase the delivery rate in a wireless ad-hoc network. The nodes use timers and acknowledgment messages to achieve fault-tolerant communication. Two different kinds of retransmission can be distinguished:

Hop-by-Hop Retransmissions. When a node forwards a data packet to a neighbor node, it expects to receive an acknowledgment from this neighbor node. Acknowledgments can either be active or passive. With passive acknowledgment, a node will send a packet to the next hop and then listen for the next hop to forward this packet. If the data packets match then the next hop has performed the passive acknowledgment. If the next hop sends an acknowledgment message back to the first node (ideally after its own forward), the acknowledgment is called active. However, the use of passive

acknowledgment significantly reduces the number of transmitted packets. Setting timers is easy, since only single hop propagation delays and packet processing times have to be considered. Typically, the transmitter makes a bounded number of trials to successfully forward the packet and drops it after this number has been exhausted. Hop-by-hop retransmission is used in IEEE 802.11, called Automatic Repeat Request (ARQ).

End-to-End Retransmissions. The source node needs to buffer the packet until an acknowledgment from the destination node arrives. Again, the number of trials made by the source node is typically bounded. However, setting timers in this case is much harder, since reasonable guesses would need knowledge on the number of hops, the per hop delay and the current cross-traffic. End-to-end retransmissions can be combined with hop-by-hop retransmissions.

Theoretically, when the number of hop-by-hop retransmissions is unbounded, the source node can free its buffer as soon as the packet is delivered successfully to the next hop.

Pure end-to-end retransmission wastes a lot of energy. If a packet is lost at the n^{th} hop, all previous $n-1$ transmissions become wasted effort. To deliver the packet to the n^{th} hop again, we need $n-1$ additional transmissions, if all $n-1$ transfers succeed. With hop-by-hop retransmission, just one retransmission can bring packet to the same point. For efficient use of the wireless channel, hop-by-hop retransmission is a very attractive choice. However, there are drawbacks when hop-by-hop retransmission is implemented with active acknowledgments, because the channel utilization decreases. Hop-by-hop retransmission is an efficient approach to deal with link-faults, but if sender and receiver node crashed simultaneously the packet is lost. Combining end-to-end and hop-by-hop retransmission may be a solution to this problem.

9.2 Multi-path Routing

Another approach to tolerate node failures is multi-path routing. Multi-path routing allows the establishment of multiple paths between source and destination node. In multi-path routing, duplicated packets are sent along the multiple paths between the source and the destination. The packet is regarded as successfully delivered, if one copy of the packets is received. The redundant messages increase the network load in the wireless ad-hoc network. Transmissions from a node along one path may interfere with transmissions from a node along another path, thereby limiting the achievable throughput. Nevertheless, multi-path routing results in higher throughput than end-to-end retransmission and has no problems with timer setting. But the bandwidth requirements of multi-hop routing are

higher.

Wireless ad-hoc networks are highly redundant networks. There are typically multiple paths between the source and the destination. Penrose [Pen99] studies the connectivity in a communication graph with uniformly and randomly distributed nodes and homogeneous communication range. Penrose shows that if the number of nodes is high enough, then with high probability if one starts with an empty graph (i.e., without communication edges) and adds the corresponding edges as the communication range increases, the resulting graph becomes k-connected at the moment it achieves a minimum node degree of k. So, for $k = 3$, the communication graph involves three disjoint paths at the moment when the communication range is large enough to achieve a node degree greater or equal than 3 at all nodes.

The multiple paths allow wireless ad-hoc networks to tolerate faulty nodes. To enable such capability, a fault-tolerant routing algorithm needs to explore the network redundancy by multi-path routing. In the greedy multi-path routing protocol, a source node forwards the message to the k best neighbor nodes, according to the distance from the destination node. Each intermediate node forwards the i^{th} received copy of the same message to the i^{th} closest neighbor to the destination. A node stops forwarding the message if there is no neighbor left with positive progress. As opposed to the single path greedy routing protocol the nodes in greedy multi-path forwarding must store information of the forwarded message for a certain amount of time. Different protocols for geographic multi-path routing are presented in the work of Lin and Stojmenovic [LS03].

Greedy multi-path routing makes the communication in wireless ad-hoc networks more robust against node faults. However, just as single-path greedy forwarding greedy multi-path forwarding is only possible until the message(s) reach node(s) where no neighbor is closer to the destination than the node itself. At such a local minimum the message is forwarded with perimeter routing. The explanation of greedy/perimeter routing is given in Section 7.2.4. Perimeter routing must be done by forwarding the message along the perimeter of a face of the planar topology. Redundant messages shared by multi-path greedy forwarding over multiple paths may be forwarded with perimeter routing along the same face. This causes the loss of redundancy.

9.3 Fault-Tolerant Perimeter Routing

The reliability of perimeter routing is also improved by hop-by-hop retransmission and end-to-end retransmission. Multi-path routing also improves the fault-

tolerance against faulty nodes. It is even possible to always use the first counter-clockwise edge for a message in perimeter mode and the first edge in clockwise order for a redundant message. Hence, the message is forwarded in both directions along the perimeter of the face.

However, the shortcoming in perimeter routing is the required planarity of the topology. Forwarding in a non-planar graph according to the principles of perimeter routing can circulate a message in a face where no node is closer to the destination node than the node where greedy routing previously failed. The message is imprisoned and circulates forever in this face. The number of circulating messages may increase with increasing lifetime until the major part of bandwidth is wasted for circulating messages and the throughput of the wireless ad-hoc network falls to zero. The simple greedy/perimeter routing protocol is unable to detect such message loops. The nodes need either additional knowledge about the topology or a more advanced routing protocol than greedy/perimeter routing must be used.

We analyze in the following chapters whether a position-based topology control algorithm is strong enough to guarantee the planarity of the topology in presence of failures.

Alles was lediglich wahrscheinlich ist, ist wahrscheinlich falsch.

Rene Descartes (1596-1650)

Chapter 10

Fault Tolerant Position-based Topology Control with Unreliable Failure Detectors

A position-based topology control algorithm like the SDT protocol presented in Chapter 6 computes a planar topology which is required by greedy/perimeter routing to reliably deliver a message to the destination node. Each node in the wireless ad-hoc network computes locally its contribution to the overall topology using only information about its single hop neighbor nodes. However, nodes can become faulty and it is essential for an appropriate construction of the topology that the status (correct of faulty) of a specific node is agreed upon by all of its neighbor nodes.

Fault tolerance is an important issue in the context of wireless ad-hoc networks. Particularly for topology control algorithms it is important to tolerate node failures, since connectivity of the entire network may depend upon a single node. Fault tolerant topology control must guarantee a "sufficiently accurate" topology to enable a fault-tolerant routing algorithm the delivery of messages. The aim of this chapter is to analyze the requirements that are necessary to tolerate *crash failures* of nodes in position-based topology control. In localized approaches, crashed nodes typically result in an inconsistency of the local contributions to the overall topology. This inconsistency can have disastrous implications on routing: A message can get into a loop or can get lost.

A totally localized routing algorithm is unable to recognize a message loop and therefore unable to guarantee message delivery. Hence, a fault tolerant topology control algorithm must avoid the occurrence of message loops. Another common problem is message loss. We assume that this problem is dealt with by a fault

tolerant routing algorithm which is able to recognize a message loss and retries to send the message.

10.1 Correctness

In general, there are two kinds of correctness properties that each algorithm must satisfy in all executions: safety and liveness. Intuitively, a safety property specifies that ”bad things” do not happen, in any configuration, and a liveness property specifies that ”good things” eventually happen in some configuration. The notions of safety and liveness properties have been first introduced by Lamport [Lam77]. Alpern and Schneider [AS85] were the first to give a formal definition of these properties. A topology control algorithm based on a proximity graph requires the following safety and liveness property:

Safety. Every state arising during the execution of the topology control and the routing algorithm is free of message loops.

Liveness. The constructed topology eventually becomes planar and and the chosen routing path is the most efficient path, according to the appropriate proximity graph and the used routing algorithm.

To achieve liveness, a *stable period* is required. A stable period is a period of time during which no node crashes or is suspected as faulty and no new nodes are added. If these requirements are fulfilled during a period which lasts long enough, the algorithm will produce the desired topology. If these requirements are not fulfilled, the algorithm does not lose safety but simply does not produce the desired topology.

The routing algorithm must always has the possibility to forward a message across the network from the source node to the destination node. The occurrence of message loops must be avoided.

10.2 System Model and Crash Failures

From an end-to-end perspective, wireless ad-hoc networks are asynchronous distributed systems. To say that a system is asynchronous is to make no timing assumptions whatsoever. It is, because of multi-hop communication, indeed impossible to give meaningful bounds on the time it takes to transmit a message

from its sender to its receiver. This model has several advantages: e.g., simplicity, portability. Unfortunately, there are also drawbacks in asynchronous systems as soon as there are failures. Because of the time freedom is it impossible for a node to determine whether another node in the network has actually crashed or is only very slow.

Chandra and Toueg [CT96] have introduced the concept of *unreliable failure detectors* to solve this dilemma in the context of the consensus problem without restricting the asynchronous model more than necessary. A failure detector can be seen as a distributed oracle related to the detection of failures. Failure detector modules monitor the system and inform the algorithm about nodes they suspect to have failed. The algorithms can run on a totally asynchronous network and completely independent from any timing assumptions.

Of course, failure detectors by itself do not allow to get rid of any synchrony assumptions. In order to implement a failure detector, some timing assumptions are usually needed. In our setting, we can conveniently exploit the fact that direct neighbor nodes in a wireless ad-hoc network typically allow time-bounded communication. This synchrony can of course be used to implement failure detectors. Any synchronous behavior is encapsulate in the implementation of the failure detectors. An introduction to the concept of unreliable failure detectors is given in Section 10.2.1.

We will show that failure detectors can be used to avoid the occurrence of message loops and to guarantee the successful operation of perimeter routing in the presence of failures. We concentrate in our analysis on permanent node crash failures. If a node crashes, the crashed node will be removed from the local topologies of the neighbor nodes and some edges, which were blocked by the crashed node, can become part of the topology. The new topology is again planar and connected. Node crashes can lead to undesired states where some neighbors suppose node s is still alive while others have already detected the crash of s. The communication network is assumed to be reliable, i.e., it does not lose or generate messages. The state of a node is correct until it crashes. A node that does not crash during the execution is said to be correct; otherwise it is faulty.

This work is, to our knowledge, the first attempt to formally define and prove the requirements for a fault tolerant topology control algorithm with failure detectors and was published in [Str05b] for the first time.

10.2.1 Failure Detectors

The essential characteristic of failure detectors is the quality of the guess they provide about failures. As defined by Chandra and Toueg, a failure detector is basically defined by two properties, namely, a *completeness property* and an *accuracy property*. Completeness is on the actual detection of failures, while accuracy restricts the mistakes a failure detector can make.

Chandra and Toueg identified multiple classes of failure detectors. All of these have in common that eventually every process that crashes is permanently suspected by every correct process (strong completeness property). Chandra and Toueg also present classes of failure detectors satisfying only a weaker variant of the completeness property ("Eventually every process that crashes is suspected by some correct process."), but as those failure detectors can be transformed into failure detectors satisfying the stronger property, we ignore these variant for our analysis. Hence, the multiple classes of failure detectors can be distinguished by the accuracy property:

Perfect ($\mathcal{P}$). No process is suspected before it crashes.

Strong ($\mathcal{S}$). Some correct process is never suspected.

Eventually Perfect ($\diamond\mathcal{P}$). There is a time after which correct processes are not suspected by any correct process.

Eventually Strong ($\diamond\mathcal{S}$). There is a time after which some correct process is never suspected by any other correct process.

Failure detectors are normally defined for fully connected networks. In sparse networks, like a wireless ad-hoc network, one can either simulate a fully connected network or use *local* failure detectors. Local failure detectors, as defined in the work of Hutle and Widder [HW05], satisfy the same properties as traditional failure detectors, but just for direct neighbors. The usage of local failure detectors is particularly suitable in our approach, because the algorithms for the computation of the local proximity graphs only have to communicate with the single hop neighbor nodes.

A very simple local failure detector implementation for wireless ad-hoc networks would be based upon timeouts. Each node periodically sends heartbeats to each of its neighbors and each neighbor waits some periods before it suspects another node. A detailed description on the implementation of local failure detectors is given in thesis of Martin Hutle [Hut05].

10.3 Fault Tolerant Topology Control with Unreliable Failure Detectors

In our approach, all the nodes in a wireless ad-hoc network decide locally whether an edge becomes part of the topology or not. The nodes make consistent decisions, if any two nodes have the same view about the common nodes in their neighborhood. The computed topology can be wrong, however, if nodes have a different opinion regarding a common neighbor node. In such a case, the stipulated properties of the topology (e.g., planarity) cannot be guaranteed. An erroneous topology has various effects on both greedy and perimeter routing.

In a wireless ad-hoc network, a failure detector can make two different kinds of mistakes. First, as a consequence of the completeness property, a faulty node can not or not yet be detected by some neighbor nodes. And second, as a result of the accuracy property, a correct node can be falsely suspected by some neighbors. Both mistakes generate inconsistencies between local topologies and can result in a non-planar topology.

In more detail, mistakes can have the following effects in the topology:

- **Fault-case 1.** A faulty node is not or not yet suspected by the failure detector of node s. Hence, the faulty node and – maybe – the edge to the faulty node is part of the local proximity graph of node s. Moreover, the faulty node can block some other edges to become part of the local proximity graph of node s.
- **Fault-case 2.** A correct node is suspected by the failure detector of a neighbor node s. Thus, node s excludes the suspected node and computes a new, wrong local proximity graph. Moreover, the exclusion of the wrongly suspected node enables the admission of additional edges. The computed overall topology can become non-planar in this case.

Obviously, both fault cases presented above cannot occur, if the failure detectors of all neighbor nodes consistently come to the same, right or wrong, decision at the same time.

We analyze in the following the consequences of these mistakes in greedy and perimeter routing. A fault tolerant topology control scheme should allow a reliable routing algorithm to detect message loss and, in particular, protect the network from circulating messages.

First, we analyze the consequences of fault-case 1 in greedy routing:

- A forwarded messages to a faulty node gets lost.

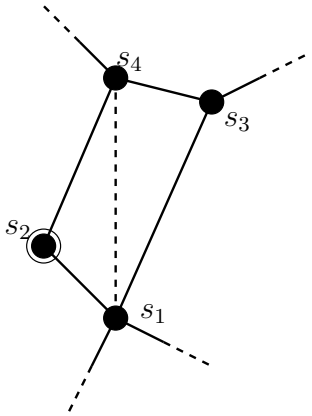


Figure 10.1: *Faulty Node* s_2

- The faulty node can block other edges to become part of the topology. The consequences for a sender node can be that a message must take a longer path than necessary. This is unpleasant, but has no dangerous effects on greedy routing. Figure 10.1 explains this issue: Node s_1 wants to send a message to node s_4, but node s_1 has not suspected the faulty node s_2 and hence edge (s_1, s_4) is not part of the local topology of node s_1. Node s_1 forwards the message to node s_3 and node s_3 onward to s_4. On the other hand, a recipient can receive the greedy message over a communication edge which is not part of the computed topology. This is inefficient and not the intention of a topology, but not dangerous in greedy routing. The receiver node forwards the message according to the routing principles. Assume node s_1 in Figure 10.1 has correctly suspected the faulty node and therefore, edge (s_1, s_4) is part of its local topology. If node s_4 has not suspected node s_2, node s_4 can receive a message directly from s_1, but edge (s_1, s_4) is not part of the local topology of node s_4.

The consequences of fault-case 1 in perimeter routing:

- The following lemma shows that an unsuspected faulty node cannot yield message loops.

  Lemma 37. *An unsuspected faulty node cannot force a perimeter message into a message loop.*

Proof. An unsuspected faulty node has two ways to affect perimeter routing. First, a correct node can forward a message to the faulty node and the message gets lost. Second, the unsuspected faulty node can block an edge to become part of the topology. Assume an edge (s_i, s_j) and a faulty node s_k within the critical section of (s_i, s_j). The following three cases can occur:

- The faulty node s_k is not or not yet suspected by the correct nodes s_i and s_j. Hence, node s_k is further on part of the local topologies of s_i and s_j. A correct node can forward a message to the faulty node and the message gets lost. The blocked edge has no affect, because the local topologies of the correct nodes are consistent. No message loop can occur.
- It can happen that one correct node (say s_i) includes the blocked edge (s_i, s_j) in its local topology, whereas node s_j use still its local topology without edge (s_i, s_j). The local topologies are inconsistent. Hence, node s_j can receive a message over a communication edge which is not part of its local topology. Node s_j forwards the message according to the routing principles of perimeter routing. The receiver node is either the faulty node s_k or a correct node. Message loops cannot occur.
- Both correct nodes have suspect the crashed node. The local topologies are consistent and no message loop can occur.

□

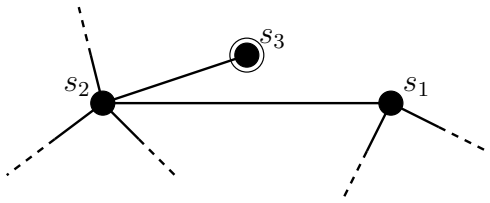


Figure 10.2: *Faulty Node* s_3

Figure 10.2 shows an example of the second case presented in Lemma 37. The failure detector of node s_2 has not or not yet suspected the faulty node s_3. Assume node s_1 has suspect node s_3 and hence, edge (s_1, s_2) becomes part of the local proximity graph of node s_1. Node s_2 has not suspected node s_3 and edge (s_1, s_2) is not part of the local proximity graph of node s_2. In perimeter mode, the message is forwarded along the perimeter of the

chosen face. Therefore, node s_1 forwards the perimeter message to node s_2 and node s_2 receives a message over an edge which is not part of its local topology. This is unsound but not dangerous. According to the perimeter mode, node s_2 forwards the message along the next edge counterclockwise. In this example, node s_2 forwards the message to the faulty node s_3 and the message gets lost, but in other examples will the message be forwarded to a correct node.

A wrongly suspected node (Case 2) has the following consequences in greedy routing:

- The wrongly suspected node is eliminated from the the local proximity graph of the sender node, a forwarded message must take a longer path than necessary or the topology becomes disconnected.
- A recipient can receive a message from the wrongly suspected node.

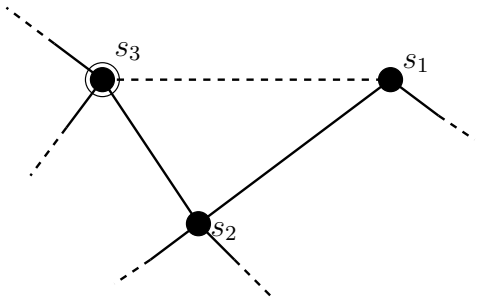


Figure 10.3: *Wrong suspected node* s_3

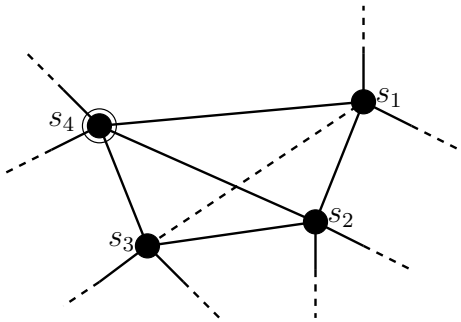


Figure 10.4: *Wrong suspected node* s_4

The consequences of fault-case 2 in perimeter routing are the following:

- A wrongly suspected node can have disastrous implications for a message in perimeter mode (see Figure 10.3). Assume node s_3 is wrongly suspected by node s_1. Hence, s_1 forwards the perimeter message to node s_2 and s_2, according to the routing principles of perimeter routing, onwards to node s_3. Since s_3 does not suspect s_1, it forwards the message to node s_1. If s_3 is still suspected by s_1, node s_1 receives a message from the wrongly suspected node. If node s_1 has reverted the suspicion in the meantime, the message circulates ongoing in the face. In a topology based on the the Delaunay triangulation (such as the Short delaunay triangulation), a wrongly suspected node has a second possibility to force a perimeter message into a message loop. Assume a system of four nodes as shown in Figure 10.4 and node s_4 is wrongly suspected by s_1. Hence, edge (s_1, s_3) is included in the local topology of node s_1 and s_1 forwards a perimeter message to node s_3. The message circulates ongoing in face s_2, s_3, s_4.

If a faulty node is not suspected (fault-case 1) by a correct node s, a message can get lost because node s sends the message to the faulty node. A message loss can occur in any network and it is the aim of a fault tolerant routing algorithm to detect a message loss (e.g., by the absence of acknowledgment messages). Node s can retry to send the message or can wait a short period of time until the fault node is suspected by the failure detector of node s and the local topology is updated. Now, node s forwards the message in the new topology according to the routing principles.

A wrongly suspected node is a correct node, forwards messages according to the routing principles, and can forward a message to the wrong suspecting node. Hence, it is not allowed to delete a suspected node immediately from the neighbor list. In contrast to fault-case 2, an unsuspected faulty node (fault-case 1) is still a faulty node and does not send messages.

A wrongly suspected node can disconnect the topology, if the edge between the suspecting and the suspected node is the only connection between two parts of the topology.

Unfortunately, a wrongly suspected node can also yield message loops. As mentioned above, a message loops has disastrous consequences, because it is impossible to detect its occurrence locally and the message circulate perpetually in the network. Hence, we must avoid the occurrence of message loops.

The disconnection of the topology, a message loop, as well as an “unknown” node, can only occur if a correct node is suspected by some neighbor nodes.

Therefore, the failure detectors must avoid wrong suspicions. Only the class of perfect failure detectors $\mathcal{P}$ fulfills this requirement (see Section 10.2.1). The other failure detector classes allow a wrong suspicion in one or another manner; temporarily by the class of $\diamond\mathcal{P}$ or permanently by the classes of $\mathcal{S}$ and $\diamond\mathcal{S}$.

We prove in the following two lemmata that the class of perfect failure detectors is strong enough to guarantee liveness and safety of greedy/perimeter routing in the presence of crash failures.

Lemma 38. *Liveness: Using $\mathcal{P}$, the topology satisfies the desired properties after a stable period.*

Proof. The accuracy property of the perfect failure detector class ensures that no node is suspected before it crashes. Furthermore, by the completeness property, eventually every node that crashes is permanently suspected by every correct node. The neighborhood sets of the nodes are eventually consistent (i.e., if a node n_i is in the neighborhood set of site n_j, then node n_j is in the neighborhood set of n). Hence, no conflicts between local proximity graphs can occur. The local proximity graphs become planar, the topology corresponds to the desired proximity graph and the routing algorithm can chose the most efficient path – according to the routing algorithm abilities. □

Lemma 39. *Safety: Using $\mathcal{P}$, the arise of a routing loop is impossible.*

Proof. Let us assume a message circulates along a face f, $f = e_1, e_2, \ldots e_k$, $k \geq 3$, of our topology, permanently forwarded by the k nodes of this face. One of these k nodes, say node s_1, has injected the message into this face or one node in face f has received the message from a node outside of the face, say from node s_0. In both cases, a correct edge of face f must have been ignored. This can only happen if one node belonging to this edge suspects the other node. In the first case, node s_1 made a wrong suspicion and in the second case s_0. By the accuracy property of a $\mathcal{P}$-class failure detector, a node will never be suspected before it crashes. □

10.4 Conclusion

We showed in this chapter that the class of perfect failure detectors is able to guarantee a sufficiently accurate topology for greedy/perimeter routing in the presence of crash failures. However, the crash failure model is very simple. A failure

model that allows erroneous or malicious nodes to send wrong position information is more severe than the crash failure model. The analysis in such a setting will be much more interesting.

Noli turbare circulos meos!

Archimedes (285–212 v.Chr.)

Chapter 11

The Network Model

The principles of proximity graphs are originated in computational geometry. Some unit has global knowledge here and computes the complete proximity graph. The computation of a topology for a wireless ad-hoc network must be done completely localized, since each node has only information about its single hop neighbor nodes. We assumed in Chapter 5 and Chapter 6 circular and homogeneous communication ranges for the computation of the local topologies, i.e., that all nodes have the same maximal communication range and each node can communicate with all nodes within this communication range. The aim of this chapter is to exactly specify the required basic assumptions and to analyze the requirements for heterogeneous wireless ad-hoc networks. For the correct computation of a position-based topology are two requirements absolutely necessary. We call this requirements *Consistency* and *Completeness*.

11.1 Assumptions

We assume that all nodes in the network have negligible difference in altitude, so they can be considered roughly in a plane. We further assume the existence of a Medium Access Control (MAC) protocol (e.g., CSMA/CA) which creates reliable point-to-point connections between the nodes in the wireless network. This forms the basic infrastructure needed for wireless hop-by-hop communication. We recapitulate in the following the definitions of Section 2.5:

Each wireless ad-hoc network can be modeled as a communication graph $CG(\mathcal{S}, \mathcal{E})$ in the plane, with a set of sites $\mathcal{S}$ and a set of edges $\mathcal{E}$. A communication edge $e = (s_i, s_j)$, $e \in \mathcal{E}$, represents a wireless link of the communication graph $CG(\mathcal{S}, \mathcal{E})$. An edge (s_i, s_j) is present in $CG(\mathcal{S}, \mathcal{E})$ if and only if $\|s_i, s_j\|$ is less than or equals the maximal communication range of node s_i. The neighbor-

hood of a node $s \in \mathcal{S}$, denoted by $\mathcal{N}(s)$, is the set of nodes within the maximal communication range of node s.

We assume a connected network for our analysis. If the network is partition-free, the greedy/perimeter routing approach delivers the message to the destination. Greedy/perimeter routing, in particular perimeter routing can be used to detect network partitions. The perimeter message will return to the node injecting the message, because the message never reach a node that is closer to the destination node.

As mentioned in Section 5, it is possible only in fully connected network to compute a topology which contains exactly the same edges as the proximity graph. It depends on the position of the sites whether an edge is present in the proximity graph or not, whereas it depends additionally on the communication range if a communication edge is used in the topology or not. Nodes in wireless ad-hoc networks can normally communicate only with a small subset of the total number of nodes in the network. It is simply impossible to use a communication edge which is longer than the communication range even if the edge is present in the proximity graph, however.

11.2 Network Model

Each node s has a *real communication area* $\overline{A}(s)$ which depends on transmission power, noise, interference, and blockages due to physical obstructions. $\overline{A}(s)$ is different at each node, irregular and generally unknown. We assume a *circular communication area* $A(s)$ with some known *communication range* $R(s)$. Hence, $A(s) = disc(s; R(s))$, the disc with center s and radius $R(s)$ (see Figure 11.1). $A(s)$ must satisfies the following condition: $A(s) \subseteq \overline{A}(s)$.

The neighborhood $\mathcal{N}(s)$ of node $s \in S$ is the set of nodes within s's circumcised communication area $A(s)$ and node s can directly communicate with every node $s_i \in \mathcal{N}(s)$ (Definition 5). A topology based on a proximity graph provides a lot of advantages. To benefit from these advantages it is necessary to include each edge in the topology which is present in the corresponding proximity graph and where the length of the edge is shorter than the communication range. The following two properties for $\mathcal{N}(s)$ are necessary to guarantee this requirement: *Consistency* and *Completeness*.

- **Consistency:** If a node s_i is in the neighborhood set of node s, then node s

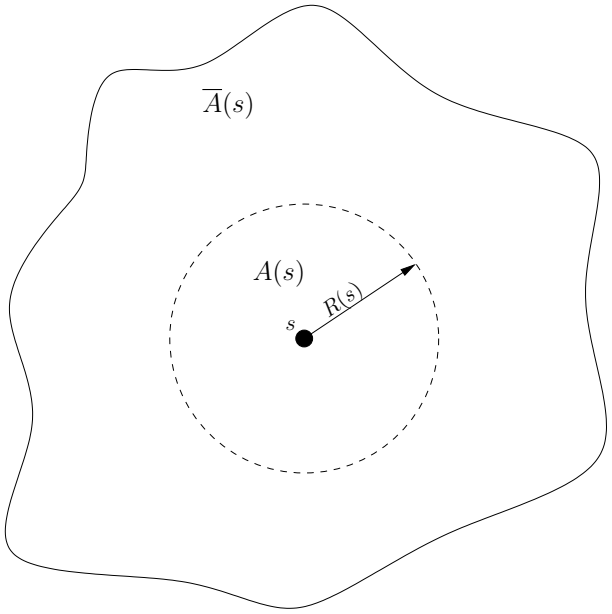


Figure 11.1: *The real and the circumcised communication area of node s.*

is in the neighborhood set of s_i.

$$\forall s_i \in \mathcal{N}(s) : s \in \mathcal{N}(s_i) \tag{11.1}$$

- **Completeness:** All nodes s_i within the circumcised communication area $A(s)$ are elements in the neighborhood set of node s.

$$\forall s_i \in A(s) : s_i \in \mathcal{N}(s) \tag{11.2}$$

11.2.1 Consistency

Communication in wireless networks depends on the transmission power of the wireless nodes and the surrounding environment. In some situations is it possible that node s_1 can receive a message from node s_2, but node s_2 cannot receive a message from node s_1. This *asymmetric communication* can happen because node s_2 use more transmission power than s_1 and achieves therefore a larger communication range, or if the communication from node s_1 to node s_2 is disturbed by stronger interference than the communication from node s_2 to node s_1.

For the correct computation of a distributed topology according to our approach is it absolutely necessary to prohibit asymmetric communication. The exchange of information between neighbors in a network with asymmetric communication allows single hop communication only in one direction. The other

direction requires, in the worst case, the broadcast of the information in the whole network until the desired neighbor node receives this information. Since the computation of the topology should be totally localized, we require direct information exchange between single hop neighbors. Multi-hop communication between neighbor nodes should be avoided. We cannot guarantee the single hop communication if the communication in the network is asymmetric. The local computation of a bi-connected topology becomes impossible if the consistency property is violated, because the local computation requires a bi-connected communication graph.

The consistency property is necessary to guarantee symmetric communication (bidirectional links). Consistency or symmetry is a standard assumption for the most common MAC (Medium Access Control) approaches. Although implementing protocols with unidirectional wireless links is technically feasible (see [BGLA01, KTC01, PHM00, Pra01, RCM02] for unidirectional link support at different layers), the actual advantage of using unidirectional links is questionable. For example, in [MD02] Marina and Das have shown that the high overhead needed to handle unidirectional links in routing protocols outweighs the benefits that they can provide, and better performance can be achieved by simply avoiding them. The high overhead is due to the fact that low level protocols, such as the MAC protocol, are naturally designed to work under the symmetric assumption. For instance, the MAC protocol defined in the IEEE 802.11 standard [Dep97] is based on RTS - CTS message exchange: when node s_i wishes to send a message to one of its neighbors, e.g., s_j (at this level, communication is only between immediate neighbors), it sends a RTS (request-to-send) to s_j, and waits for a CTS (clear-to-send) message from s_j. If the CTS message in not received within a certain time, then message transmission is aborted and it is tried again after a backoff interval. Hence, tor the protocol to work s_i must be within the communication range of s_j and vica versa, i.e., the communication must be symmetric.

11.2.2 Completeness

Consistency on its own is not sufficient to guarantee the computation of a position-based topology. It is additionally necessary to guarantee that a node which can communicate with a neighbor node, can also communicate with all nodes which are closer to the node than the neighbor node. Hence, if a node s_1 can communicate with a node s_2, node s_1 must also be able to communicate with all nodes $s_i \in S$ with $\|(s_1, s_i)\| \leq \|(s_1, s_2)\|$. We call this property completeness. The following examples show the importance of the completeness property for the computation of a position-based topology.

Euclidean minimum spanning tree.

An edge (s_i, s_j) is not included in the localized computed Euclidean minimum spanning tree if the edge is neither part of local Euclidean minimum spanning tree $T_{EMST}(\mathcal{N}(s_i), EMST(\mathcal{N}(s_i)))$ of node s_i nor part of the local Euclidean minimum spanning tree $T_{EMST}(\mathcal{N}(s_j), EMST(\mathcal{N}(s_j)))$ of node s_j. Assume a set of three nodes $S = \{s_1, s_2, s_3\}$ arranged as shown in Figure 11.2(a) where it is impossible for s_2 and s_3 to communicate with each other. Node s_1 assumes an edge between node s_2 and node s_3 in $T_{EMST}(\mathcal{N}(s_1), EMST(\mathcal{N}(s_1)))$. However, the localized computed Euclidean minimum spanning tree $T_{EMST}(S, E)$ shown in Figure 11.2(b) is only uni-connected (see Section 3.1), because there exists no path to node s_3.

(a) $CG(\mathcal{S}, \mathcal{E})$ (b) $T_{EMST}(S, E)$

Figure 11.2: *The importance of the completeness property in the Euclidean minimum spanning tree.*

Relative neighborhood graph, Gabriel graph, and Delaunay triangulation.

A communication edge is unsuitable for a topology, if the critical section of the edge contains a further node. The critical sections of Relative neighborhood graph, Gabriel graph, and Delaunay triangulation can be seen in Chapter 4. If there is a wireless node s_3 in the critical section of an edge (s_1, s_2) and neither node s_1 nor node s_2 can communicate with node s_3, the edge (s_1, s_2) will be included in the topology even if the empty critical section requirement is not fulfilled. Such wrong included, additional edges can destroy the planarity of the topology which decreases the performance of the perimeter routing algorithm. Routing loops cannot occur in this situation because the additional edge is bidirectional: The edges in a planar graph form faces. An additional edge either divides a face in two smaller faces or connects two nodes of two different faces. The planarity is only destroyed in the second case. The perimeter routing algorithm forwards the message along the face of a planar graph. Is the perimeter message

forwarded along an illegal edge, an additional edge of the second case as mentioned before, the perimeter message is also forwarded along the edges of the added face and is returned over the additional edge to the original face. The additional way of the perimeter message can be very long, which impacts the performance of perimeter routing, but the message eventually arrives at the destination node.

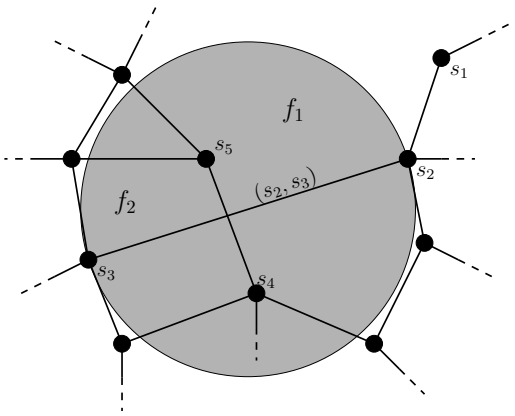


Figure 11.3: *Example to show the must of the completeness property.*

Figure 11.3 shows an example of the above situation. The presented topology uses the Gabriel graph as proximity graph. We assume in this example that it is impossible to communicate for two nodes s_2 and s_3 with node s_4 and node s_5. Hence for both nodes the critical section of edge s_2, s_3 is empty and both nodes affiliate the edge to the topology. Assume node s_1 forwards a message in perimeter mode to node s_2. According to the routing principles of perimeter routing, node s_2 forwards the message along the illegal edge (s_2, s_3) to node s_3 and thereby directly in the adjacent face f_2. In face f_2, the message is forwarded clockwise, over the edge (s_4, s_5) and back to node s_3. Node s_3 sends the message over edge (s_2, s_3) to node s_2 and this node forwards the message along the face f_1. Finally, the message is still on the right way, but such an excursion to another face takes longer time and can decrease the performance of the routing algorithm dramatically.

Figure 11.3 is also a good example for the importance of the formerly proposed consistency property: Assume additionally to the assumptions presented above that node s_3 cannot communicate with node s_2. Hence, edge (s_2, s_3) is an unidirectional edge orientated from node s_2 to node s_3. If node s_2 forwards a message along edge (s_2, s_3) to node s_3, it is impossible for

the message to leave face f_2 and the message circulates continuously in the network. The same can happen if it is impossible only for node s_2 to communicate with node s_4 and node s_5. Node s_3 recognizes now that there are other nodes in the critical section of edge (s_2, s_3) and does not affiliate the edge to the topology. Node s_2 cannot communicate with node s_4 and node s_5 and so edge (s_2, s_3) is again an unidirectional edge, however. A perimeter message sent from node s_2 to node s_3 circulates along the perimeter of face f_2.

A solution to this problem can be that the corresponding nodes of an edge inform each other about potential nodes in their critical section to get a common view about their critical section. Node s_3 informs node s_2 about two nodes, s_4 and s_5 lying in the critical section of edge (s_2, s_3). Unfortunately, the result will be that node s_3 also ignores edge (s_2, s_3) in its contribution to the topology. This can yield to partitions in the topology, because the nodes are not necessarily connected over another path. Partitions of the topology are unacceptable if the communication graph is connected. The network shown in Figure 11.3 is still connected, but it is impossible to guarantee this property locally in the general case.

Yao graph.
The completeness property is primarily important to guarantee the planarity of the localized computed topology. However, the Yao graph is not a planar topology and so the completeness property has not the same importance as in the other topologies. The completeness property is not required in $T_{YG_k}(S, E)$ to guarantee connectivity as in the Euclidean minimum spanning tree, either because according to Definition 12, the Yao graph additionally contains the reverse edges of the directed edges. It suffices therefore to communicate with only one node in a cone and not with all (nearer) nodes. The existence of the reverse edges is guaranteed by the consistency property. The localized computed Yao graph and all other localized computed topologies presented in Section 5 requires the completeness property to guarantee the stretch factors of the topologies (Section 5.3), however.

This examples explain the necessity of the completeness property. To stay localized, to avoid message loops, and to guarantee the bi-connectivity of position-based topologies it is required that each node can communicate with all nodes within its communication range.

Unfortunately, completeness cannot be achieved by multi-hop communication, because it is impossible to know how many hops are required to reach all

nodes within a critical section. Some global knowledge will be necessary to achieve completeness in a neighborhood and to guarantee multi-hop communication to the nodes in the neighborhood, but this is not a localized approach. Completeness must be a priori guaranteed by the chosen communication ranges.

11.2.3 Homogeneous/Heterogeneous Communication Ranges

The consistency as well as the completeness property, are indispensable conditions for the localized computation of topologies based on proximity graphs. Both (11.2) and (11.1) are implied by the following property:

$$\forall s_i \in disc(s; R(s)) : s \in disc(s_i; R(s_i)) \tag{11.3}$$

Equation 11.3 is fulfilled by setting $R(s) = R$ for all s, when $R = \max\{R' : disc(s, R') \subseteq \overline{A}(s)\ for\ all\ s\}$ is the minimum of the communication ranges of all nodes.

A wireless ad-hoc network with a common communication range at all nodes is called homogeneous (Definition 6). Such a homogeneous network obviously fulfills the desired completeness and consistency property. However, homogeneous communication ranges are not required for the localized computation of position-based topologies. A wireless ad-hoc network with heterogeneous communication ranges can also be used to compute a topology with the quality properties presented in Chapter 5, but the chosen communication ranges must fulfill the consistency and the completeness property.

The lemmata and proofs presented in Chapter 5 do not require homogeneous communication ranges. The satisfaction of the completeness and the consistency property is sufficient for the correctness of the lemmata in Chapter 5.

11.2.4 Unit disk graph

A special form of a homogeneous network model is the *Unit disk graph*. In the Unit disk graph are all communication ranges normalized to one unit. The Unit disk graph is a widely employed model for the study of topology control (e.g., [GGH$^+$01b, LCWW03]) and routing (e.g., [BMSU01, KWZ02, KWZ03b]) in wireless ad-hoc networks. Completeness and Consistency are obviously fulfilled.

Definition 29 (Unit disk graph). *The Unit disk graph $UDG(S)$ of a set S is defined as an bidirectional graph, where there is an edge between two sites s_i,s_j if and only if the Euclidean distance between s_i and s_j is at most 1.*

11.3 Remarks on Localized Computed Topologies

The accurate specification of the network model gives us the basics for addressing some questions left open in previous chapters.

11.3.1 Computation of $T_{EMST}(S, E)$ in Heterogeneous Wireless ad-hoc Networks.

The localized computed Euclidean minimum spanning tree $T_{EMST}(S, E)$ is a special case, because it is the only topology that contains the global topology $EMST(S)$ as subgraph. This is only true if the communication ranges are homogeneous (the connectivity of the communication graph is a precondition). In a wireless ad-hoc network with heterogeneous communication ranges, it is possible that the complete Euclidean minimum spanning tree is not part of the communication graph. However, the spanning tree of the communication graph with shortest Euclidean length is a subgraph of the localized computed topology and ensures the connectivity of $T_{EMST}(S, E)$ (Lemma 12).

11.3.2 Global Topologies versus Localized Computed Topologies

As mentioned in Chapter 6, the Short delaunay triangulation is not necessarily a subgraph of the global Delaunay triangulation. $T_{SDT}(S, E)$ contains all edges of the global Delaunay triangulation $DT(S)$ that are shorter than the communication range and some additional edges which are not present in $DT(S)$. We analyze in the following the reasons for this fact. $T_{RNG}(S, E)$, $T_{GG}(S, E)$, and $T_{YG_k}(S, E)$ are obviously subgraphs of the respective global graphs. We assume in Chapter 5 homogeneous communication ranges, the following lemma gives a detailed proof according to the completeness and the consistency property.

Lemma 40. *$T_{RNG}(S, E) \subseteq RNG(S)$, $T_{GG}(S, E) \subseteq GG(S)$, and $T_{YG_k}(S, E) \subseteq YG_k(S)$*

Proof. No point p in the critical section of an edge (s_i, s_j) in the Relative neighborhood graph (resp.Gabriel graph) lies farther away from node s_i than node s_j. More precisely, $\|(s_i, p)\| < \|(s_i, s_j)\|$ for all points p of the critical section of edge (s_i, s_j) in the Relative neighborhood graph and the Gabriel graph. Hence, each node s_i which is able to communicate with node s_j can, because of the completeness property, also communicate with all nodes within the critical sections edge (s_i, s_j). The consistency property guarantees that node s_j can also communicate with all nodes within the critical section of edge (s_i, s_j) (in Relative neighborhood

graph and Gabriel graph). Hence, both nodes have a consistent and complete view of their common critical section and need no information exchange for the computation of a topology based on Relative neighborhood graph or Gabriel graph. Each node is completely informed about the critical section of the edges originated at it and computes therefore its local contribution to the topology correctly. The localized computed Relative neighborhood graph $T_{RNG}(S, E)$ and the localized computed Gabriel graph $T_{GG}(S, E)$ are subgraphs of the global proximity graphs ($RNG(S)$ and $GG(S)$).

The completeness property assures that a node s_i in the Yao graph is able to screen its critical sections completely, because no point in a critical section (e.g., the critical section of the directed edge (s_i, s_j)) lies farther away from node s_i than node s_j. The localized computed Yao graph $T_{YG_k}(S, E)$ contains all edges of the global Yao graph and is furthermore a subgraph of $YG_k(S)$. □

A little bit more complicated is the situation for the computation of the Delaunay triangulation. A point p of the critical section of an edge (s_i, s_j) in the Delaunay triangulation can be farther away from s_i and/or s_j than s_i from s_j. In such a situation is it impossible for a node s_i to decide on its own whether an edge originated at s_i is valid in the topology or not. It is furthermore impossible for a localized computation algorithm of a topology based on the Delaunay triangulation to know how much information from how many nodes is necessary to gain a complete view of a critical section. It is still impossible for nodes with limited communication range to check the critical section of a Delaunay edge completely. However, information from the single hop neighbor nodes are sufficient to compute a planar topology [GGH+01b,LCWW03,Str05a]. The completeness property ensures that at least one node is able to communicate with the other three nodes if two communication edges cross (i.e. the topology is non-planar). This node can inform the three other nodes about the immanent non-planarity. This is the construction principle of the Short delaunay triangulation protocol presented in Chapter 6. The impossibility to supervise the complete critical section of an edge yields to additional edges with are not part of the global Delaunay triangulation, but this additional edges do not destroy the planarity of the topology.

11.4 Weakness of Position-based Topology Control

The main weakness of the network model required for the localized computation of a planar topology is the assumption that the radio coverage area is a perfect circle (completeness property). This assumption is quite realistic in open air flat environments, but it is critical in indoor or urban scenarios, where the presence of

objects, walls, buildings, and so on, renders the radio coverage area extremely irregular. Further, the area and shape of the radio coverage is influenced by weather conditions and by the interference with pre-existing infrastructure (e.g., power lines, base stations, and so on). Including all these details in the network model would make it extremely complicated and scenario-dependent, hampering the derivation of meaningful and sufficiently general analytical results. For this reason, the network model described above, although quite simplistic, is widely used in the analysis of wireless ad-hoc networks.

However, for some topologies a weaker completeness property can be used. Starting from the restrictive Unit disk graph model, Barrière et al. [BFNO01] as well as Kuhn et al. [KWZ03a] present a less restrictive and more realistic network model. Two nodes are connected by an edge if their distance is less than or equal to d, d being a parameter between 0 and 1. Furthermore, if the distance between two nodes is greater than 1, there is no edge between them. In the range between d and 1 the existence of an edge is not specified. Is it possible to guarantee the localized computation of a position based topology with this network model? Does this network model guarantee the required connectivity and planarity? The answer is yes, but only for topologies based on the Gabriel graph [KWZ03a, BFNO01]. Two nodes s_i and s_j are consistently informed about the critical section of edge (s_i, s_j) if $d \geq 1/\sqrt{2}$. The topology is correctly computed, if the nodes inform each other about other nodes in their common critical section. Unfortunately for the computation of the Short delaunay triangulation[1],it is absolutely necessary to be able to communicate with all nodes within the communication range. If two edges cross it is required that at least for one of the four involved nodes can communicate with the three other nodes (Lemma 27). Only the completeness property guarantees this requirement.

[1]And for any other localized computed and planar topology based on the Delaunay triangulation

Jeder Narr kann Regeln aufstellen
und alle Narren werden sich
danach richten.

Henry D. Thoreau (1817–1862)

Chapter 12

Virtual Positions and Moving Nodes

Each position-based algorithm, no matter whether routing or topology control, requires the node positions to satisfy its task. However, all position services known until today are imprecise. Even the most accurate position service can make a small and varying position error. Using this estimated positions, the nodes form a virtual network which is slightly different from the real network. Position-based algorithms must tolerate the inaccuracies and must compute a correct topology even with time-varying virtual positions. An introduction to position services is given in Section 12.1, and Section 12.2 presents the requirements for the correct computation of the topology with imprecise position information.

As shown in Chapter 10, position-based topology control algorithms can be made tolerant to crash failures. However, node crashes are not the only causes of topology changes in a network. Moving nodes and hence changing node positions are the dominant causes of topology changes. Common position-based algorithms assume that all nodes obtain changes in the position information of a neighbor node immediately and at the same time. This assumption is unsustainable in realistic settings. Hence, the position of a node can differ in the view of the neighbor nodes for a short period of time. The aim of Section 12.3 is to verify if moving nodes impair the message delivery of greedy/perimeter routing, i.e., whether moving nodes could cause message loops.

12.1 Position Service

Position-based routing as well as position-based topology control require a position service that provides all network participants (the nodes) with their location, e.g., by using the Global Position Service (GPS), or by triangulation with the help of fixed beacons or the positions of neighbor nodes [NN01, SRL02]. A survey of

position services can be found in [HB01]. Position information enables context-sensitive computing and is the basis of many location specific applications, such as service discovery, resource discovery and mapping. Location knowledge is also an important information for supervision and security features. In wireless ad-hoc networks the position is in many cases more important than a specific node ID. For example, for tracking applications it is more interesting where the target is located than in the ID of the reporting node.

Position information is, for various reasons, an interesting feature for communication in wireless ad-hoc networks. As detailed in previous chapters, position-based routing algorithms (e.g., greedy/perimeter routing) require for message delivery the positions of the neighbors and the position of the destination node. The position of nodes is furthermore an indispensable requirement for the computation of a planar topology and hence for the usage of perimeter routing.

In this chapter, we assume that the nodes in the network as well as the position service work correctly, i.e., the deviation from the real position is not larger than the inaccuracy of the position service, a node does not crash, and each node faithfully sends its acquired position to the neighbor nodes. In particular, no node sends wrong positions to some or all of its neighbor nodes. The problem of tolerating crash failures in position-based topology control is analyzed in Chapter 10.

12.2 Virtual Topology

We assume that nodes and position service are fault free, but suffer from some inaccuracy. Hence, the acquired position of a node, we call it *virtual position* in the following, is slightly different from the *real position*. The gap between virtual and real position is at most the inaccuracy of the position service, denoted by Δ. We analyze in the following the consequences of these inaccuracies in position-based topology control.

The computed *virtual* positions of the nodes only approximate the *real* positions. Because of the inaccuracy of the position service, the resulting graph is called the virtual communication graph of the network. A topology based on the virtual positions is not necessarily identical to the topology computed on the real positions. The virtual topology satisfies exactly the same properties as the real topology, but only if the communication ranges of the nodes in the virtual network satisfies the same properties as the communication areas of the nodes in the real network. The required properties are consistency and completeness (see Chapter 11). The communication range of the nodes must be extended by $2 * \Delta$,

because the distance between two virtual positions can be at most $2 * \Delta$ larger than the distance between the real positions of these two nodes. The extended communication range is denoted by $\overline{R}(s)$, $\overline{R}(s) = R(s) + 2 * \Delta$ and is shown in Figure 12.1.

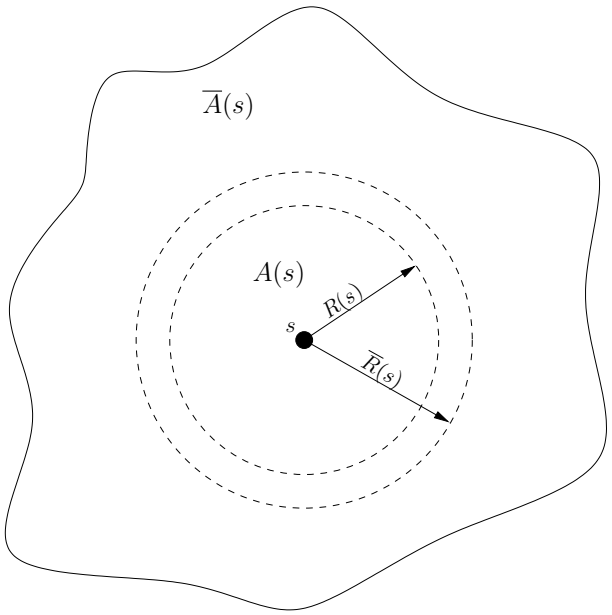


Figure 12.1: *Communication Range to tolerate position inaccuracies.*

Figure 12.2 gives an illustration of the requirements for guaranteeing consistency and completeness in presence of position inaccuracies. For simplicity, we assume in this example a homogeneous network with a common communication range R and also a common extended communication range $\overline{R}$. The extension of the communication range to $R + 2 * \Delta$ is sufficient to guarantee consistency. More complicated is the situation to guarantee completeness, however. The virtual position of a node can lie within the communication range of a specific node whereas the distance to the real position could be larger than the communication range. Figure 12.2 shows a Relative neighborhood graph. $\overline{s}_3$ and $\overline{s}_5$ are the virtual positions of node s_3 and node s_5. We assume for simplicity that the virtual position of s_1, s_2 and s_4 are identical with the real positions of these nodes. If node s_1 can only communicate with all nodes within communication range R, node s_3 violates the construction rule of the topology. The virtual position of node s_3 lies within communication range R of node s_1 in Figure 12.2. Hence, the completeness property is violated because the distance to $\overline{s}_3$ is shorter than the distance to s_2. Each node must communicate with all nodes within communication range $\overline{R}$

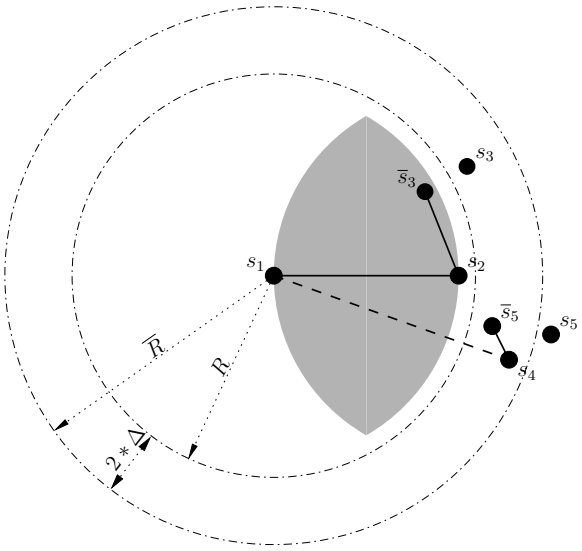


Figure 12.2: *Virtual Positions. The shaded region is the critical section of a Relative neighborhood graph*

to guarantee completeness. However, it is not allowed for a node to involve the nodes located in the annulus formed by R and $\overline{R}$ in the computation of its local contribution of the topology. In Figure 12.2 is it impossible for node s_1 to communicate with node s_5, because the real position of n_5 is not within the extended communication range of node s_1. If node s_1 involves node s_4 in the computation of the topology, it must also involve all nodes that are closer to node s_1 than node s_4 (Completeness property). However, the virtual position of node s_5 can be nearer to s_1 than node s_4. The completeness property will be not fulfilled. Hence it is necessary to involve only nodes in the computation of the topology with are nearer than communication range R.

12.3 Position Updates

Current position-based algorithms usually assume that all affected nodes acquire the changes of node positions in the wireless ad-hoc network simultaneously . However, in reality a certain time passes until all relevant nodes receive a new node position. This time interval can be very long, e.g., if the new position must be broadcasted in the whole network to all other nodes. Hence, algorithms which only have to inform the single hop neighbor nodes are considerably faster and more efficient. The position-based algorithms presented in Chapter 5 and Chapter 6 for topology control, and in Chapter 8 for routing are totally localized, i.e., they require only position information of the single hop neighborhood. Hence,

position updates can be done very efficient. However, short time delays until all neighbor nodes receive the new node position are still inevitable. These time delays can yield to inconsistencies discrepancies between neighbor nodes; i.e., one node may already have computed a new local topology with the updated position while another node still uses the old topology.

Before we can examine the effects of such discrepancies, we must consider the procedure of a position update. Each node broadcasts its position to the neighbor nodes, if it departs a certain distance from its previous position. This distance Δ_1 must also be considered in the communication range, because the completeness as well as the consistency property must be continuously maintained. However, two nodes can move in opposite directions and so the total distance must be considered twice. The required new communication range is denoted by $\overline{\overline{R}}(s)$, $\overline{\overline{R}}(s) = R(s) + 2 * (\Delta + \Delta_1)$. The extended communication range can be seen in Figure 12.3.

Another approach for position updating is periodic sending. Each node broadcasts is position after a certain time period. The first approach sends only message if a node departs a certain distance from its previous position. Hence, the total interference in the wireless network is lower. Furthermore, periodic updating requires an estimation of the maximum moving speed of wireless nodes to determine Δ_1.

The time difference between the reception of a position update message at the first neighbor node and the reception of the update message at the last neighbor node in a message broadcast to the single hop neighbor nodes seems to be very short. A lot of algorithms uses broadcasts to distribute a message to all nodes within the communication range. This looks reasonable at the first glance, but broadcast are in many instances only a construct for the simpler description of algorithms. If a node broadcasts a message to all of its single hop neighbor nodes and all of these nodes send an acknowledgment message at the same time back to the broadcasting node, the messages interfere. Therefore, at least the Medium Access Control (MAC) protocol divides each broadcast into unicast messages. Due to the partitioning of the broadcast into a number of unicast messages, the time difference between the reception of the message at the first and at the last single hop neighbor node depends not only on the distance from the sending node to the nodes in the neighborhood, but also on the number of single hop neighbor nodes.

The different update receive times yield, for a short time, discrepancies between the local topologies. Some nodes had received the new position of the

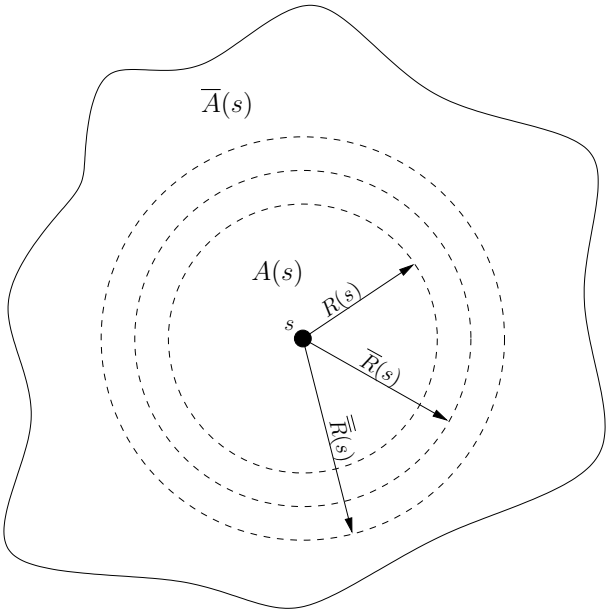


Figure 12.3: *Communication Range to tolerate position inaccuracies and node movement.*

single hop neighbor node and computed a new topology, whereas the other nodes still use the old topology. If the proximity graph chosen for the construction of the topology is the Delaunay triangulation, then problems in perimeter routing can occur. The Delaunay triangulation is the densest graph of the three presented proximity graphs and partitions the plane into triangles. If two triangles form a convex quadrilateral, a minimal position change of one node can perform an *edge flip* on the shared edge. The edge flip occurs if the critical section of the shared edge is violated by another node. In the left quadrilateral of Figure 12.4 the critical section of edge (s_2, s_4) is empty and hence is the edge part of the Delaunay triangulation. If node s_2 moves from its position in the left quadrilateral to its new position in the right quadrilateral, then the critical section of edge (s_2, s_4) is no longer empty. The dashed circle represents the critical section of edge (s_2, s_4) through node s_1. The empty circle property is violated, see Definition 14. The edge cannot be part of the topology any more. After the node movement the circle through s_1, s_3 and s_4 is empty[1] and edge (s_1, s_3) becomes part of the topology.

An edge flip on its own is no problem for the routing algorithm, but a situation where an edge is also nonexistent because it is longer than the communication range can be a problem for the perimeter routing algorithm. Figure 12.5 shows an

[1]Consequently the circle through s_1, s_2 and s_3 is also empty.

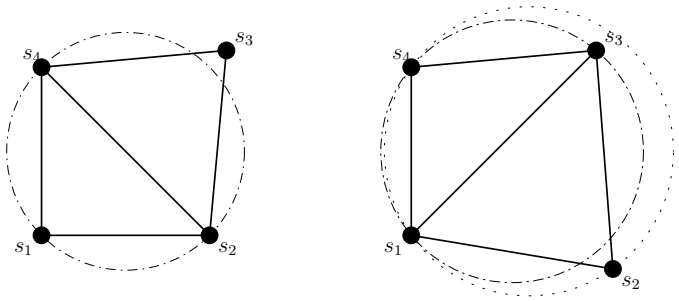


Figure 12.4: *Edge flip in the Delaunay triangulation.*

example. The distance between node s_3 and node s_4 is larger than the communication range and hence there is no edge between s_3 and s_4 in the topology. We assume that node s_2 moves to a new position and thus edge (s_1, s_3) is removed from the topology and edge (s_2, s_4) must be added to the topology. Figure 12.5 shows only one example. Please note that an edge flip can occur by any movement of only one of the four nodes of the quadrilateral. Routing loops can occur in two different scenarios:

Case 1. Node s_3 uses still the old position for the computation of the topology, whereas node s_1 and node s_4 use already the new position. Hence s_3 forwards the message to node s_1, s_1 further to s_4 and node s_4 back to node s_2. The perimeter message circulates, even after s_3 also updates its local topology.

Case 2. s_2 use the old topology, s_1, s_3 and s_4 the new computed topology. The perimeter message is forwarded from node s_3 to s_2, from node s_2 over s_1 to node s_4 and from node s_4 back to node s_2. The message circulates even if node s_2 change in the following to the new topology.

Hence, a node can forward a perimeter message to a node which does not lie — after the computation of the new topology — on the perimeter of the face. In Case 1 node s_1 received the perimeter message over an edge which is no longer part of the topology. Obviously, the node can recognize this occurrence locally, but it is impossible for a node to return the message to the correct face without additional knowledge about the topology. Hence, the message is undeliverable if the routing algorithm is totally localized (e.g., as the greedy/perimeter routing approach presented in Chapter 8). The only solution is to restart the greedy/perimeter routing but there is no guarantee that the same thing does not happen again. In the second case is it generally impossible to detect the occurrence of the routing loop without additional knowledge about the topology or the used routing path.

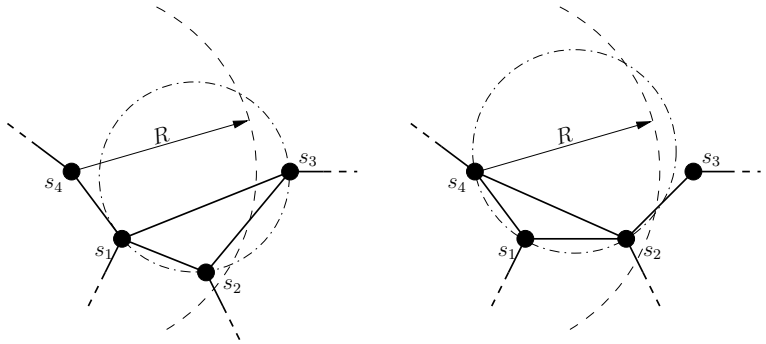


Figure 12.5: *Example of an edge flip in a topology based on the Delaunay triangulation.*

12.4 Unreliability of Perimeter Routing

One could argue that the occurrence of the two cases presented above is rare, because the update of the single hop neighbor nodes is fast enough and the number of perimeter messages is comparatively small. Furthermore, this problem exists only in topologies based on the Delaunay triangulation. We due could hence use a planar topology based upon another proximity graph. However, the problem of captured messages can also occur in other situations:

- If two nodes, which were outside of each others communication range, converge and suddenly add their common edge to the topology.
- If a new node joins the network and enables a bridge between two nodes which are outside of each others communication range.

In both cases, the face will be divided into two parts. If there is a perimeter message in the first part of the previous face and the node which starts the perimeter mode in the other face, it may be impossible for the perimeter message to leave the new face. It can happen that the perimeter message never reach a node closer to the destination node than the node injecting the perimeter message, and hence, the message circulates endlessly in the new face. This scenario is not only relevant to a topology based on the Delaunay triangulation, this situation can happen to all planar topologies. Figure 12.6 shows an example:

Node s_1 wants to send a message to node s_2. At node s_3 is it impossible to forward the message in greedy mode and so the message is forwarded in perimeter mode. If the perimeter message passes s_4 and node s_5 moves a little bit closer to node s_4[2], the edge (s_4, s_5) is added to the topology and the perimeter message is

[2]Or a new node between s_4 and s_5 joins the network.

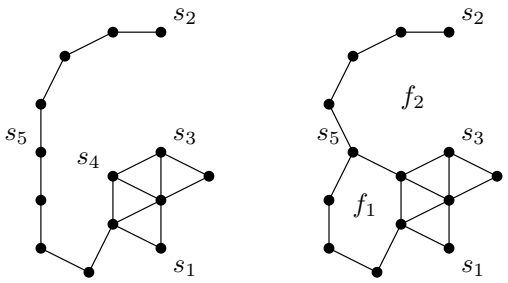


Figure 12.6: *Captured message in a topology based on the Gabriel graph.*

captured in face f_1.

The routing loop can neither be detected locally nor can this situation be solved locally. There are only two possible solutions to detect such loops: The message header either stores the chosen path completely or the nodes store for each perimeter message the predecessor and the successor. Both methods are, especially in large networks, very inefficient.

The simple greedy/perimeter routing approach is, because of its locality, unable to guarantee the message delivery between arbitrary nodes, even in a network environment with faultless nodes and perfect communication links, if nodes are allowed to move or to join the network. The nodes need either additional knowledge about the topology or a more reliable routing protocol than greedy/perimeter routing.

Die Zukunft soll man nicht
voraussehen wollen, sondern
möglich machen.

Antoine de Saint-Exupéry
(1900–1944)

Chapter 13

Conclusion

This PhD-Thesis considers two interesting and important problems of wireless ad-hoc networks: topology control and routing. We present an overview of different techniques for topology control and give a detailed analysis of position-based topology control. Various approaches to routing are also considered and again, the focus of the overview is on position-based approaches. The thesis discusses the fault tolerance of position-based communication against crashed nodes and its behavior in the context of moving nodes. Furthermore, the presented SDT/VAR approach propose an efficient common implementation of topology control and routing. The *Short delaunay triangulation* (SDT) is a powerful topology based on the construction rules of the *Delaunay triangulation* and the *Voronoi-aided routing* (VAR) protocol is an efficient implementation of the well known *greedy/ perimeter routing* approach.

The Short delaunay triangulation (SDT) algorithm computes an underlying topology for greedy/perimeter routing in wireless ad-hoc networks. The topology fulfills the required planarity and is the densest possible planar topology. The density of a topology is important to guarantee the existence of efficient paths between the nodes and an equal distribution of network load. The Voronoi-aided routing (VAR) algorithm uses the same data structure than the topology for efficient forwarding of a message to a destination. Both algorithms are totally localized. They use only information of the single hop neighbor nodes for the computation of the topology resp. to forward a message. Local computation is almost the most important property to algorithms in wireless ad-hoc networks. It guarantees the scalability of the network, because a localized algorithm is completely independent of the total number of nodes in the network.

A common implementation of topology control and routing like the SDT/VAR approach is the most efficient implementation of greedy/perimeter routing. The

computation of the topology has time complexity $O(n \log n)$, where n only is the number of single hop neighbor nodes in the communication graph. This matches the lower bound on the construction of a planar topology. The routing algorithm uses the same data structure and only requires $O(\log n')$ time complexity to find the next node in the communication path, where $n' \leq n$ is the number of single hop neighbor nodes in the topology. This number can be significantly lower than the number of single hop neighbor nodes in the communication graph. The number of single hop neighbor nodes is on average 6 in the Delaunay triangulation.

Greedy forwarding is a very simple and efficient routing approach, but it requires a recovery strategy to escape from a local minimum. Our recovery strategy, perimeter routing, is also a very simple, but it requires a planar underlying topology to forward a message, only with local knowledge, towards the destination. The computation of each planar topology for perimeter routing requires completeness and consistency (as defined in Section 11.2) of the communication range of each node in the wireless ad-hoc network. These properties must be a priori guaranteed by the underlying wireless ad-hoc network, because it is impossible to guarantee these properties locally. However, completeness and consistency need not be guaranteed for the maximal communication range. It suffices that each node guarantees these properties for an arbitrary communication range. Nevertheless, the required completeness and consistency property restrict the usability of perimeter routing.

Another handicap for greedy/perimeter routing is the vulnerability of perimeter routing to moving nodes and new joining nodes (Section 12.4). Each change in the topology can force a message into a message loop. The primary problem is not that messages are circulating in the network and wasting resources. The primary problem is that the most fundamental requirement of routing algorithms cannot be guaranteed – the delivery of messages. It is impossible for the simple greedy/perimeter routing approach to detect a message loop. Each routing approach that uses a planar underlying subgraph (e.g., *Compass routing* [KSU99], *GPSR* [KK00], *Face routing* [BMSU01], *AFR* [KWZ02], ...) requires a solution to this problem, or must be comfortable with an almost reliable routing algorithm. The solution can be a more complex routing approach. Either the nodes or the message headers have to store previous traffic to detect the occurrence of a message loop, but both methods decrease the scalability of greedy/perimeter routing.

What are potential application areas for a greedy/perimeter routing approach like SDT/VAR?

The construction rules of proximity graphs facilitate the completely localized

construction of a planar topology, independent from the total number of nodes in the network, but with the drawback that the delivery of messages cannot be guaranteed in a wireless ad-hoc networks with mobile nodes and new joining nodes. Hence, application areas for the presented SDT/VAR approach are non-mobile wireless networks like mesh or rooftop networks (Section 2.2.2). These networks have to cover wide metropolitan areas. Scalability and fault tolerance against malfunctioning stations are the most basic requirements of mesh networks.

Mesh networks are, because of non-mobile stations, not vulnerable to mobility and also the problem of new joining nodes can be solved in mesh or rooftop networks. The stations in mesh networks are more powerful than nodes in mobile ad-hoc networks (Section 2.2.1) or sensor networks (Section 2.2.3). Hence, each station can use its own GPS-receiver and no distributed position service is required. GPS offers the station time information and a common time base can be used to handle new joining nodes: The node that changes the message to perimeter mode adds additionally to its own position the current time to the message header. If a new station appears in the network, the new station sends its position and the current time to the neighbor stations and each neighbor station computes a new topology with the new station. If a neighbor station receives a perimeter message with an older timestamp than the timestamp of the new station, the neighbor station forwards the perimeter message on the old topology. If the timestamp of the message is younger, the message is forwarded on the new topology. The new station computes its own local topology with the position of the neighbor stations and will be increasingly integrated in the topology. This approach can only be used to handle new joining nodes, because ignoring new stations cannot disconnect the network. However, it is not possible to guarantee the connectivity of a network if moving nodes are ignored.

The installation of wired infrastructure requires a lot of time and is, especially in urban areas, very expansive. Mesh networks represent a promising alternative solution. Our approach guarantees the scalability in mesh networks and enables efficient communication with a localized routing approach. A new station must only communicate with the neighbor stations, no login, or registration is required, regardless if there are ten or ten million stations in the network. Failures of stations can easily be handled, because only the neighbor stations must react. The greedy/perimeter routing approach guarantees that no routing paths or routing tables must be updated as required with other routing approaches.

A self-configurable communication approach, like our SDT/VAR protocol, for hundreds of thousands of stations in a metropolitan area represents a significant improvement in wireless network technology.

Irren ist menschlich. Aber wenn
man richtig Mist bauen will,
braucht man einen Computer.

Dan Rather, CBS-Fernsehreporter

Appendix A

Delaunay Triangulation Algorithms

The topology for a wireless ad-hoc network is not computed at one node for the whole network but distributed. Each node computes localized its own contribution to the overall topology, only with information about the single hop neighbor nodes. Nevertheless it is important to use an efficient and flexible algorithm for the computation of the local topology. We presented in Chapter 5 five topologies based on different proximity graphs. The focus of this chapter is only on the computation of the Delaunay triangulation because of the following three reasons:

- The Delaunay triangulation is planar and the densest planar graph.
- Euclidean minimum spanning tree, Relative neighborhood graph, and Gabriel graph are subgraphs of the Delaunay triangulation and often computed from the Delaunay triangulation.
- The Delaunay triangulation and the Voronoi diagram are dual graphs. We use the Delaunay triangulation in the Short delaunay triangulation protocol presented in Chapter 6 and the Voronoi diagram in the Voronoi-aided routing protocol presented in Chapter 8. Hence, it is more important for us to analyze the different algorithms for the computation of Delaunay triangulation (resp. Voronoi diagram) than for the computation of other proximity graphs.

The Delaunay triangulation and its dual, the Voronoi diagram, belong to the classical subjects in computational geometry. For various reasons, it is much more convenient to express algorithms in terms of the Delaunay triangulation than the Voronoi diagram. One reason is to avoid manipulating computed values: because of arithmetic inaccuracies in computations, Voronoi vertexs may be poorly determined if the defining sites are close to being affinely dependent. Another reason is that it is simpler to manipulate a cell complex with regular bounded cells than

one with irregular unbounded cells. Hence, we concentrate in the remaining paper on the computation of the Delaunay triangulation. The Voronoi diagram can be obtained in linear time from the Delaunay triangulation.

Five basic classes of algorithms are proposed in the literature for constructing the Delaunay triangulation:

- Incremental Insertion [Bow81, GS78, Wat81]
- Gift wrapping [Dwy91]
- Divide and Conquer [GS85]
- Plane-sweep [For87]
- Convex Hull

Algorithms for the computation of the Delaunay triangulation were usually designed for good performance on a given static set of sites. Nodes in a wireless ad-hoc network are generally not static; new nodes join the network, nodes leave the network (because of power constraints or errors) and mobile nodes move around and change the connectivity of the network. Rebuilding the topology after each change in the network is very inefficient. Hence, successive maintenance of the topology may performs better in a dynamic environment. The next section gives an overview of the algorithms mentioned above, and subsequently, Section A.2 presents methods to extend these algorithms for the usability in the context of dynamic wireless ad-hoc networks.

A.1 Static Delaunay Triangulation Algorithms

This section outlines some of the fundamental algorithms that are known for Delaunay triangulations. We only give a rough overview of the algorithms; for more general surveys, Aurenhammer [Aur91], Fortune [For95] and Su et al. [SD95] are recommended.

A.1.1 Incremental Insertion Algorithm

The simplest and most intuitive class of algorithms for constructing the Delaunay triangulation are the incremental insertion algorithms. These algorithms add sites one by one and update the diagram after each site is added. For simplicity, start with an initial triangle so that all subsequent sites lie inside this triangle.

Suppose, $R := \{s_0, s_1, \ldots s_{m-1}\}$, $R \subset S$, is a set of sites, $DT(R)$ is the Delaunay triangulation of this site set, and $s_m \in S \setminus R$ is the next site to add. The overall structure of the algorithm is as follows:

1. Determine the triangle $\Theta \in DT(R)$ containing the new site s_m.

2. Update the triangulation $DT(R)$ to $DT(R \cup \{s_m\})$.

An important requirement of the incremental insertion algorithm is an efficient data structure to determine the triangle containing s_m.

A well known data structure is the *Delaunay Tree* introduced by Boissonnat and Teillaud [BT86, BT93]. The Delaunay Tree stores all successive versions of the Delaunay triangulations during the insertion process and maintains adjacency relationships between the triangles of the current triangulation. The internal nodes are triangles that have been deleted or subdivided at some point in the construction, and the current Delaunay triangulation is stored at the leaves. For each step of location, the algorithm moves from the triangle containing the site at one level to one of a constant number of triangles that might contain the site at the next level. This structure yields an expected $O(\log n)$ complexity for the location of a single site, if the sites are inserted in random order. The total expected cost of the location algorithm is therefore $O(n \log n)$. The expected memory complexity of the Delaunay Tree is $O(n)$ [BT93].

Another data structure is the *Quad-Edge* structure of Guibas and Stolfi [GS85]. This structure is simpler and stores only the current triangulation. Location is performed by starting at a random edge and walking across the triangulation in the direction of the new site until the enclosing triangle is found. The expected complexity for the location of a single site is $O(\sqrt{n})$. The big advantage of Quad-edge is the simultaneous representation of the Delaunay triangulation and the Voronoi diagram in the same data structure.

Updating can be performed in two different ways: The first approach, introduced by Green and Sibson [GS78], is based upon edge flipping. The second approach, introduced simultaneously by Bowyer [Bow81] and Watson [Wat81], replaces the affected polygon.

Flipping

When a new site s_m is inserted, the three vertices of the enclosing triangle Θ are connected to s_m.[1] (We now have a triangulation again, but not necessarily a Delaunay triangulation.) Next, a recursive procedure tests for each edge e of Θ whether it is still a valid Delaunay edge. For this is suffices to test whether s_m lies in the circumcircle of the triangle $\overline{\Theta}$ lying on the other side of e. If not, no update of the triangulation beyond e is necessary. Otherwise we flip the diagonal of the convex quadrilateral formed by Θ and $\overline{\Theta}$, and examine the two other edges of $\overline{\Theta}$ for validity. We continue this way until all edges that we encounter are valid, and then we stop.

Lemma 41. *Let e be an edge of a triangulation of S. Either e is a Delaunay edge, or e is flippable and the edge created by flipping e is a Delaunay edge.*

Proof. Let s_i and s_j be the sites opposite to an edge e, which together with e define a quadrilateral. Let C be the circle that passes through s_i and the endpoints of e. Either s_j is strictly inside C, or s_j lies on or outside C. If s_j is on or outside C, then e is a Delaunay edge. If s_j is inside C, then the quadrilateral is constrained to be strictly convex, so the edge e is flippable. Furthermore the circle that passes through s_i and s_j, and is tangent to C at s_i, does not enclose the endpoints of e. Hence. the edge $s_i s_j$ is a Delaunay edge. □

In the worst case we require $O(n^2)$ tests and edge flips, because it is possible to construct a set of sites and insertion order where inserting the k^{th} site into the diagram causes $\Theta(k)$ updates. However, if the sites are inserted in a random order, Guibas, Knuth and Sharir [GKS92] show that the expected number of edge flips is linear no matter how they are distributed. However, adding the sites in coordinate-sorted order is mostly impossible, because the sites may not all be available at the beginning of the algorithm (especially in a wireless ad-hoc network).

Replacing

In the replacing algorithm, when a new site s_m is inserted, each triangle whose circumcircle encloses the new site is no longer a Delaunay Triangle, and is thus deleted. All other triangles remain Delaunay Triangles, and are left undisturbed. The union of the deleted triangles form a polygon P, which is left vacant by the deletion of these triangles. This polygon is replaced by a new triangulation where all vertices of P are connected to the new site s_m. All new edges created by the

[1] If s_m falls on an edge $e \in DT(R)$, Θ is a quadrilateral formed by the two triangles sharing e. Edge e is deleted and s_m is connected to the four vertices of this quadrilateral.

insertion of a site s_m are Delaunay edges and have s_m as an endpoint.

This new edges are Delaunay edge due to the following simple lemma:

Lemma 42. *Let s_m be a newly inserted site, let Θ be a triangle that is deleted because its circumcircle encloses s_m, and let s_i be a vertex of Θ. Then $s_i s_m$ is a Delaunay edge.*

Proof. The circumcircle of Θ encloses no vertex but s_m. Let C be the circle that passes through s_i and s_m, and is tangent to the circumcircle of Θ at s_i. C is empty, so $s_i s_m$ is a Delaunay edge. □

All new edges created by the insertion of a site s_m have s_m as an endpoint. This must be true, because if an edge (not having s_m as endpoint) is no Delaunay edge before s_m is inserted, it will not be a Delaunay edge after s_m is inserted.

In the worst case, the complexity of this algorithm is $O(n^2)$, because the circumcircles of all triangles can contain the new site. On average, finding the triangles whose circumcircle contains a new site takes constant time, and for all sites time $O(n)$.

A.1.2 Gift-Wrapping

Another class of Delaunay triangulation algorithms constructs the Delaunay triangulation by starting with a single Delaunay Triangle and discovering the remaining triangles one by one. Say that an edge of a Delaunay Triangle is *unfinished* if the algorithm has not yet identified the other Delaunay Triangle that shares the edge. To *finish* the edge is to find the other triangle. (If the edge lies on the boundary of the Convex Hull of the input, the edge becomes finished when the algorithm recognizes that there is no other adjoining triangle.)

The gift-wrapping algorithm maintains a list of unfinished edges, which initially contains the three edges of the first Delaunay Triangle. The basic approach is as follows:

1. Remove an arbitrary unfinished edge e from the list.
2. Choose a candidate site s for the third vertex of the new triangle Θ. Other sites are tested to see if they fall within the circumcircle of triangle Θ. If one does, it becomes a new candidate site s. When a site s is found such that the circumcircle of Θ contains no other sites then Θ is added to the Delaunay triangulation. If no site can be found, then e lies on the boundary of the Convex Hull.

3. Check each edge of Θ, except e, against the list. If an edge is already present in the list, then the edge is now finished, so remove it from the list. Otherwise the edge is new, so insert it into the list.

The running time of this algorithm is $O(nn_\Theta)$, where n is the number of input sites and n_Θ is the number of triangles in the output. However, the time to find a new Θ can be improved, if the sites are uniformly distributed. Dwyer [Dwy91] presents a sophisticated site search algorithm that considers on average only a constant number of sites for finding a new triangle. Hence, the entire Delaunay triangulation may be constructed in $O(n_\Theta)$ average time complexity.

A.1.3 Divide-and-Conquer

The divide-and-conquer algorithm was the first worst-case optimal algorithm for computing the Delaunay triangulation. Guibas and Stolfi give a careful description of the algorithm in [GS85]. The Delaunay triangulation is constructed as follows:

1. The sites are sorted lexicographically by the x-coordinate (ties resolved by the y-coordinate).

2. If there are three or fewer sites, the Delaunay triangulation is computed directly. Otherwise, the sites are divided into two halves of approximately the same size and the Delaunay triangulation of each half is recursively computed.

3. The two triangulations are merged.

Step 2 and Step 3 are recursively applied to construct the whole Delaunay triangulation.

Merge the two triangulation starts with the lower common tangent and work upwards through the diagram. Triangles are deleted from the first half, if their circumcircles contain sites of the other half (and visa versa) and new edges connecting the two halves are generated. The merge step takes linear time in the worst case, so the worst-case running time of the algorithm on n sites is $O(n \log n)$.

A.1.4 Plane-sweep

The plane-sweep algorithm [For87] constructs the Delaunay triangulation by using an imaginary line, the *sweepline*, that sweeps across the plane. The sweepline divides the plane into two halves, where the area behind the sweepline has already

been triangulated. The area before it still waits to be processed, however.

Let the sweepline l_y be a horizontal line with y-coordinate y. l_y can be divided into intervals, where each interval $I_y(s)$ corresponds to a site s. Each point $p \in I_y(s)$ is the topmost point of a circle which touches site s. At an endpoint q of $I_y(s)$, the circle also touches a second site t with interval $I_y(t)$ (q is also the endpoint of interval $I_y(t)$). The sweepline moves upwards, and changes the existing triangulation in two different ways. First, l_y can encounter a new site u. Assume that u lies in interval $I_y(s)$, a new edge from s to u is added to the triangulation and for slightly larger y, u becomes an interval $I_y(u)$ on the sweepline. Second, increasing y shrinks an interval (e.g., $I_y(s)$) to one point p. The circle with topmost point p touches three sites (e.g, u, s and t), because the endpoints of the subsequent intervals are situated in point p. A new edge is added from u to t and completes the Delaunay Triangle ust. The interval $I_y(s)$ disappears from the sweepline when increasing y.

The algorithm maintains the sweepline and an event queue ordered by y-coordinate. The events are passing a new site and passing the topmost point of the circumcircle of a Delaunay Triangle. The running time of the algorithm is $O(n \log n)$. The sweepline moves $O(n)$ times, because the number of events is equal to the number of sites plus the number of Delaunay Triangles. Additionally, it costs $O(\log n)$ time to maintain the event queue and the sweepline.

A.1.5 Convex Hull

There is a remarkable relationship between the Delaunay triangulation in dimension d and convex hulls in $d+1$ dimensions. Each site in dimension d is appropriately mapped to a point in dimension $d+1$. The convex hull of these lifted points is computed and the projection of the upward-facing convex hull faces back to d dimension is the Delaunay triangulation of S.

In two dimensions, the connection between the two structures is the paraboloid $z = x^2 + y^2$. If we map each point (x_i, y_i) to $(x_i, y_i, x_i^2 + y_i^2)$, the plane is mapped to a paraboloid in three dimensions, and every circle C in the plane is mapped to a plane cutting the paraboloid. The portion of the paraboloid below the plane is projected inside the circle C and the portion above is projected outside. In particular, let $p, q, r \in S$, and let p', q', r' denote the projections of these points onto the paraboloid. Then $p'q'r'$ defines a face of the lower convex hull of S if and only if pqr is a triangle of the DT of S.

This allows any convex-hull algorithm to be used to construct the Delaunay triangulation.

A.1.6 Performance

Algorithm	average case complexity	worst case complexity
Incremental Insertion[2](flipping)	$O(n \log n)$	$O(n^2 \log n)$
Incremental Insertion[2](replacing)	$O(n \log n)$	$O(n^2 \log n)$
Gift-wrapping[3]	$O(n_\Theta)$	$O(nn_\Theta)$
Divide-and-Conquer	$O(n \log n)$	$O(n \log n)$
Plane-sweep	$O(n \log n)$	$O(n \log n)$

Table A.1: Complexities of Delaunay triangulation Algorithms.

Table A.1 lists the complexities of the above presented computation algorithms. We assume the usage of the Delaunay Tree [BT86, BT93] as data structure for the two incremental insertion algorithms (see Section A.1.1). Hence, the site location complexity of the algorithms is $O(n \log n)$, if the sites are inserted in random order.

A comparison of Delaunay triangulation algorithms in [SD95] shows that the divide-and-conquer algorithm is fastest and that the sweep-line algorithm comes second. The incremental algorithm performs poorly, spending most of its time in site location.

A.2 Dynamic Delaunay Triangulation Algorithms

In many situations where the Delaunay triangulation or Voronoi diagram are required, the set of sites and the position of sites are not fixed. In particular, the topology in wireless ad-hoc networks changes through node motion, appearance of new nodes and disappearance of erroneous nodes. It is often more efficient to maintain the existing Delaunay triangulation than compute the whole Delaunay triangulation again. Kim and Hoffmann [KH04] present dynamic updates of the Delaunay triangulation in the context of situation awareness in military applications.

[2]The expected location complexity is $O(\log n)$.

[3]$O(n_\Theta)$ is the number of triangles in the Delaunay triangulation, $n_\Theta = O(n)$

A.2.1 Deletion and Insertion

The simplest approach for updating the Delaunay triangulation is successive *deletion* and *insertion*. When a node has moved, we simply delete the site from the Delaunay triangulation and reinsert the site at the new position. We propose in the previous section two algorithms for insertion: flipping and replacing. Pathological cases can occur in which a single site insertion (without location) takes $O(n)$ time in two dimensions. However, such cases occur only occasionally, and it is customary to observe that the average time complexity by insertion is a constant.

The algorithm proposed by Devillers et.al [DMT92] needs $O(\log n)$ expected time to insert a site in the Delaunay triangulation and $O(\log \log n)$ expected time to delete a site. This algorithm is based on the replacing approach (see Section A.1.1). Devillers et al. use the Delaunay Tree which was introduced by Boissonnat and Teillaud [BT86, BT93]. When a site s is removed from the Delaunay triangulation, all triangles of the tree incident to s (leaves and internal nodes) must be removed and the tree must be restored if s had never been inserted. The complexity results of the algorithm hold provided that any order of insertion of the sites is equally likely, and any site is equally likely to be deleted. A detailed description and analysis of the algorithms can be found in [DMT92].

Another algorithm, based on flipping (see Section A.1.1), works as follows:

For deletion of site s, the neighboring sites of s are examined. These sites form a star-shaped polygon P with boundary $s_0, s_1, \ldots s_{k-1}, s_0$. If the line segment s_i, s_{i+2} of three consecutive sites s_i, s_{i+1} and s_{i+2} is inside P, and the circumcircle of this "potential triangle" is empty of other sites except s, then the edge s, s_{i+1} is flipped and the procedure is restarted with the same boundary except site s_i. When only three sites remain on the boundary, the algorithm terminates and site s can be deleted. Devillers [Dev99] proposes an algorithm following this approach with complexity $O(k \log k)$ (k is the node degree of the deleted site).

Again, the bottleneck of these algorithms is the location method. When the position information is updated frequently, the movement of nodes has spatial coherence in wireless ad-hoc networks. We can exploit this fact to reduce the computation needed for locating the site. The point from which the site was deleted, can be used as starting point to find the triangle encloses the new position of the site.

A.2.2 Adaptation of the Delaunay triangulation during motion

The flipping approach can also be used to transform an arbitrary triangulation into a Delaunay triangulation:

1. Determine an arbitrary triangulation of the site set.

2. Compute the Delaunay triangulation by flipping all non Delaunay edges to Delaunay edges. (see Section A.1.1)

Converting an arbitrary triangulation to Delaunay using this technique requires $O(n^2)$ flips. However, when the Delaunay triangulation already exists and the sites move only slightly, the triangulation is not destroyed and an adaptation of the Delaunay triangulation is possible. Furthermore, the flipping process should take much less than $O(n^2)$ time complexity, because most of the edges are already Delaunay edges. Adaptation becomes problematic, however, if the triangulation is destroyed by site movement. An *orientation test* is needed to verify such situations: Assume a site s of the Delaunay triangulation has moved from position p_1 to position p_2. The neighboring sites of s in the original Delaunay triangulation form a star-shaped polygon with boundary $s_0, s_1, \ldots s_{k-1}, s_0$. Assume that the line segments s_i, s_{i+1} are oriented in counterclockwise direction. If point p_2 is contained in the left halfspace of each line segment s_i, s_{i+1}, then the triangulation is valid and the flipping algorithm can be started to recompute the Delaunay triangulation. Otherwise, the triangulation is destroyed, s must be removed from position p_1 and reinserted at position p_2.

A.2.3 Kinetic Data Structure

Another method to maintain moving sites in the Delaunay triangulation are *Kinetic Data Structures*, introduced in [BGH97]. A Kinetic Data Structure contains a set of *certificates* that constitutes a proof of the property that is to be maintained or monitored. For site movement, it is assumed that each site follows a particular *flight plan*. The flight plans are functional forms to predict the future positions of all sites. They can be either computed algebraically or estimated by interpolation. With these flight plans, it is possible to calculate the time at which the certificates become false. All these "failure times" (called *events*) are kept in an event queue. Maintaining a Kinetic Data Structures is appropriately updating the structure and the event queue, event by event.

For Delaunay triangulation, the certificates are the in-circle test for each triangle. Processing an event includes recomputing of the – now invalid – Delaunay triangulation and appropriately updating the event queue, since some certificates

change through the new triangles.

A big problem of Kinetic Data Structures is the assumption of flight plans for the sites. Especially in wireless ad-hoc networks, it is nearly impossible to predict future positions as a functional form. However, Guibas and Russel [GR04] propose a method to interpolate the trajectories between the old set of positions and the new set of positions. The Delaunay triangulation is appropriately updated after each position update. The remaining problem is the frequency of position updates. If the updates are too often, a lot of computation power is wasted. Otherwise, illegal triangulation exists between the updates.

A.3 Conclusion

The presented static algorithms for the computation of the Delaunay triangulation (Section A.1) are not practical for the usage in dynamic environments. Moreover, gift-wrapping, divide-and-conquer and plane-sweep must know the whole set of sites in advance and it is impossible for these approaches to react on the arise of a new site. The only solution is perpetual recomputation of the whole Delaunay triangulation. Only the incremental insertion algorithm is able to integrate a new site in a existing Delaunay triangulation. All algorithms must be enhanced by the deletion and insertion approach (Section A.2.1) as well as the adaptation approach (Section A.2.2) to be applicable for the usage in dynamic environments. The deletion and insertion approach can efficiently be used to react on the appearance of new nodes and to eliminate erroneous nodes from the Delaunay triangulation. Adaptation is a promising approach to deal with moving nodes.

The incremental insertion algorithm based on edge flipping is the most appropriate approach for wireless ad-hoc networks. Deletion and insertion can be implemented with flipping and adaptation must be implemented with flipping. Only this concept is practical for all challenges in dynamic environments, it is very efficient because no mixture of different approaches is required, however.

Bibliography

[74984] Standard ISO 7498. Information processing systems - open systems interconnection - basic reference model, 1984.

[AMS94] Sunil Arya, David M. Mount, and Michiel H. M. Smid. Randomized and deterministic algorithms for geometric spanners of small diameter. In *Proceedings of the Thirtyfifth annual IEEE Symposium on Foundations of Computer Science*, pages 703–712, 1994.

[AS85] Bowen Alpern and Fred B. Schneider. Defining liveness. *Information Processing Letters*, 21(4):181–185, October 1985.

[Aur91] Franz Aurenhammer. Voronoi diagrams – a survey of a fundamental geometric data structure. *ACM Computing Surveys*, 23(3):345–405, September 1991.

[BCSW98] Stefano Basagni, Imrich Chlamtac, Violet R. Syrotiuk, and Barry A. Woodward. A distance routing effect algorithm for mobility (DREAM). In *Proceedings of the Fourth annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, pages 76–84, Dallas, Texas, USA, October 1998.

[BFNO01] Lali Barrière, Pierre Fraignaud, Lata Narayanan, and Jaroslav Opatrny. Robust position-based routing in wireless ad hoc networks with irregular transmission ranges. In *Proceedings of the Fifth ACM International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, pages 19–27, Italy, July 2001.

[BGB02] Ljubica Blazevic, Silvia Giordano, and Jean-Yves Le Boudec. Self organized terminode routing. *Journal of Cluster Computing*, 5(2):205–218, April 2002.

[BGH97] Julien Basch, Leonidas J. Guibas, and John Hershberger. Data structures for mobile data. In *Proceedings of the Eighth Annual*

ACM-SIAM Symposium on Discrete Algorithms, pages 747–756, 1997.

[BGLA01] Lichun Bao and Jose Joaquin Garcia-Luna-Aceves. Channel access scheduling in ad hoc networks with unidirectional links. In *Proceedings of the Fifth International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, pages 9–18, Rome, Italy, 2001.

[BGLA03] Lichun Bao and Jose Joaquin Garcia-Luna-Aceves. Topology management in ad hoc networks. In *Proceedings of the Fourth ACM International Symposium on Mobile ad hoc Networking & Computing (MobiHoc)*, pages 129–140, Annapolis, Maryland, USA, 2003.

[Blu] The Bluetooth specification is available at http://www.bluetooth.com.

[BM99] Prosenjit Bose and Pat Morin. Online routing in triangulations. In *Proceedings of the Tenth International Symposium on Algorithms and Computation (ISAAC)*, pages 113–122, January 1999.

[BMJ+98] Josh Broch, David A. Maltz, David B. Johnson, Yih-Chun Hu, and Jorjeta Jetcheva. A performance comparison of multi-hop wireless ad hoc network routing protocols. In *Proceedings of the Fourth International Conference on Mobile Computing and Networking*, pages 85–97, Dallas, Texas, USA, October 1998.

[BMSU01] Prosenjit Bose, Pat Morin, Ivan Stojmenovic, and Jorge Urrutia. Routing with guaranteed delivery in ad hoc wireless networks. *Wireless Networks*, 7(6):609–616, November 2001.

[Bow81] Adrian Bowyer. Computing dirichlet tessellations. *The Computer Journal*, 24(2):162–166, 1981.

[BT86] Jean-Daniel Boissonnat and Monique Teillaud. A hierarchical representation of objects: The delaunay tree. In *Proceedings of the Second Annual ACM Symposium on Computational Geometry*, pages 260–268, Yorktown Heights, USA, June 1986.

[BT93] Jean-Daniel Boissonnat and Monique Teillaud. On the randomized construction of the delaunay tree. *Theoretical Computer Science*, 112(2):339–354, 1993.

[BvRWZ04] Martin Burkhart, Pascal von Rickenbach, Roger Wattenhofer, and Aaron Zollinger. Does topology control reduce interference? In *Proceedings of the Fifth ACM international Symposium on Mobile ad hoc Networking and Computing (MobiHoc)*, pages 9–19, 2004.

[CG98] Tsu-Wei Chen and Mario Gerla. Global state routing: A new routing scheme for ad-hoc wireless networks. In *Proceedings of the IEEE International Communications Conference*, pages 171–175, Atlanta, Georgia, USA, June 1998.

[CM01] Douglas S.J. De Couto and Robert Morris. Location proxies and intermediate node forwarding for practical geographic forwarding. Technical Report MIT-LCS-TR-824, MIT Laboratory for Computer Science, June 2001.

[CPS99] Andrea E. F. Clementi, Paolo Penna, and Riccardo Silvestri. Hardness results for the power range assignmet problem in packet radio networks. In *Proceedings of the Third International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (RANDOM-APPROX)*, pages 197–208, 1999.

[CT76] David Cheriton and Robert E. Tarjan. Finding minimum spanning trees. *SIAM Journal of Computing*, 5(4):724–742, December 1976.

[CT96] Tushar Deepak Chandra and Sam Toueg. Unreliable failure detectors for reliable distributed systems. *Journal of the ACM*, 43(2):225–267, March 1996.

[CY02] Stephen Carter and Alec Yasinsac. Secure position aided ad hoc routing protocol. In *Proceedings of the IASTED International Conference on Communications and Computer Networks (CCN)*, November 2002.

[Del32] Boris Nikolajewitsch Delaunay. Neue Darstellung der geometrischen Krystallographie. *Zeitschrift für Krystallographie*, 84:109–149, 1932. in German.

[Del34] Boris Nikolajewitsch Delaunay. Sur la sphère vide. *Izvestia Akademia Nauk SSSR, VII Seria, Otdelenie Matematicheskii i Estestvennyka Nauk*, 7:793–800, 1934. in Russian.

[Dep97] IEEE Standard Department. IEEE 802.11, Wireless LAN: IEEE Standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, 1997.

[Dep02] IEEE Standard Department. IEEE 802.15, Wireless PAN: IEEE Standard for Wireless Personal Area Networks (WPAN), 2002.

[Dev99] Olivier Devillers. On deletion in delaunay triangulations. In *Proceedings of the Fifteenth annual Symposium on Computational Geometry*, pages 181–188, Miami Beach, Florida, USA, June 1999.

[DM05] Arindam K. Das and Mehran Mesbahi. K-node connected power efficient topologies in wireless networks: A semidefinite programming approach. Technical Report UWEETR-2005-0004, University of Washington, Department of Electrical Engineering, February 2005.

[DMT92] Olivier Devillers, Stefan Meiser, and Monique Teillaud. Fully dynamic delaunay triangulation in logarithmic expected time per operation. *Computational Geometry Theory and Applications*, 2(2):55–80, 1992.

[DPH05] Saumitra M. Das, Himabindu Pucha, and Y. Charlie Hu. Performance comparison of scalable location services for geographic ad hoc routing. In *Proceedings of the Twentythird annual Joint Conference of the IEEE Computer and Communications Societies (IEEE INFOCOM)*, Miami, Florida, USA, March 2005.

[Dwy91] Rex A. Dwyer. Higher-dimensional voronoi diagrams in linear expected time. *Discrete & Computational Geometry*, 6(4):343–367, 1991.

[Ede87] Herbert Edelsbrunner. *Algorithms in Combinatorial Geometry*. Springer-Verlag New York, Inc., 1987.

[EGS86] Herbert Edelsbrunner, Lionidas J Guibas, and Jorge Stolfi. Optimal point location in a monotone subdivision. *SIAM Journal on Computing*, 15(2):317–340, May 1986.

[FBS02] Massimo Franceschetti, Jehoshua Bruck, and Leonard J. Schulman. Microcellular systems, random walks, and wave propagation. In *Proceedings of the IEEE International Symposium on Antennas and Propagation*, June 2002.

[Fin87] Gregory G. Finn. Routing and addressing problems in large metropolitan-scale internetworks. Technical Report ISI Research Resport ISI/RR-87-180, Information Sciences Institute, University of Southern California, March 1987.

[For87] Steven Fortune. A sweepline algorithm for voronoi diagrams. *Algorithmica*, 2:153–174, 1987.

[For95] Steven Fortune. Voronoi diagrams and delaunay triangulations. In Ding-Zhu Du and Frank Hwang, editors, *Computing in Euclidean Geometry*, volume 4 of *Lecture Notes Series on Computing*, pages 225–265. World Scientific, Singapore, second edition, 1995.

[GGH+01a] Jie Gao, Leonidas Guibas, John Hershberger, Li Zhang, and An Zhu. Discrete mobile centers. In *Proceedings of the Seventeenth annual Symposium on Computational Geometry*, pages 188–196, Medford, Massachusetts, USA, 2001.

[GGH+01b] Jie Gao, Leonidas J. Guibas, John Hershberger, Li Zhang, and An Zhu. Geometric spanner for routing in mobile networks. In *Proceedings of the Second ACM International Symposium on Mobile Ad-hoc Networking and Computing (MobiHoc)*, pages 45–55, Long Beach, California, USA, October 2001.

[GH99] Silvia Giordano and Maher Hamdi. Mobility management: The virtual home region. Technical Report SSC/1999/037, Ecole Polytechnique Federale de Lausanne (EPFL), Institute for Computer Communications and Applications, October 1999.

[GK00] Piyush Gupta and P.R. Kumar. The capacity of wireless networks. *IEEE Transaction on Information Theory*, 46(2):388–404, March 2000.

[GKS92] Leonidas J. Guibas, Donald E. Knuth, and Micha Sharir. Randomized incremental construction of delaunay and voronoi diagrams. *Algorithmica*, 7(4):381–413, 1992.

[GR04] Leonidas Guibas and Daniel Russel. An empirical comparison of techniques for updating delaunay triangulations. In *Proceedings of the Twentieth annual Symposium on Computational Geometry*, pages 170–179, Brooklyn, New York, USA, 2004.

[GS69] K. Ruben Gabriel and Robert R. Sokal. A new statistical approach to geographic variation analysis. *Systematic Zoology*, 18:259–278, 1969.

[GS78] Peter J. Green and Robin Sibson. Computing dirichlet tessellations in the plane. *The Computer Journal*, 21(2):168–173, May 1978.

[GS85] Leonidas Guibas and Jorge Stolfi. Primitives for the manipulation of general subdivisions and the computation of voronoi. *ACM Transactions on Graphics*, 4(2):74–123, 1985.

[GSB03] Silvia Giordano, Ivan Stojmenovic, and Ljubica Blazevic. Position based routing algorithms for ad hoc networks: A taxonomy. In Xiuzhen Cheng, Xiao Huang, and Ding-Zhu Du, editors, *Ad Hoc Wireless Networking*. Kluwer Academic Publisher, 2003.

[HB01] Jeffrey Hightower and Gaetano Borriello. Location systems for ubiquitous computing. *IEEE Computer Magazine*, 34(8):57–66, August 2001.

[HHB+03] Tian He, Chengdu Huang, Brian M. Blum, John A. Stankovic, and Tarek Abdelzaher. Range-free localization schemes for large scale sensor networks. In *Proceedings of the Ninth annual International Conference on Mobile Computing and Networking (MobiCom)*, pages 81–95, San Diego, CA, USA, 2003.

[HL86] Ting-Chao Hou and Victor O. K. Li. Transmission range control in multihop packet radio networks. *IEEE Transactions on Communications*, 34(1):38–44, January 1986.

[HL99] Zygmunt J. Haas and Ben Liang. Ad hoc mobility management with uniform quorum systems. *IEEE/ACM Transactions on Networking (TON)*, 7(2):228–240, April 1999.

[Hut05] Martin Hutle. *Failure Detection in Sparse Networks*. PhD thesis, Vienna University of Technology, 2005.

[HW05] Martin Hutle and Josef Widder. On the possibility and the impossibility of message-driven self-stabilizing failure detection. In *Proceedings of the Seventh International Symposium on Self Stabilizing Systems (SSS)*, Barcelona, Spain, October 2005. (to appear).

[JM96] David B. Johnson and David A. Maltz. Dynamic source routing in ad hoc wireless networks. In Tomasz Imielinski and Hank Korth, editors, *Mobile Computing*, chapter 5, pages 153–181. Kluwer Academic Publishers, 1996.

[JPS01] Rahul Jain, Anuj Puri, and Raja Sengupta. Geographical routing using partial information for wireless ad hoc networks. *IEEE Personal Communications*, 8(1):1–10, February 2001.

[JRS02] Lujun Jia, Rajmohan Rajaraman, and Torsten Suel. An efficient distributed algorithm for constructing small dominating sets. *Distributed Computing*, 15(4):193–205, 2002.

[JT87] John Jubin and Janet D. Tornow. The DARPA packet radio network protocol. *Proceedings of the IEEE*, 75(1):21–32, January 1987.

[KG92] J. Mark Keil and Carl A. Gutwin. Classes of graphs which approximate the complete euclidean graph. *Discrete & Computational Geometry*, 7(1):13–28, 1992.

[KH04] Young J. Kim and Christoph M. Hoffmann. Dynamic proximity calculations for situation awareness. *Naval Research Logistics*, 51(2):166–192, March 2004.

[Kir83] David G. Kirkpatrick. Optimal search in planar subdivisions. *SIAM Journal on Computing*, 12(1):28–35, February 1983.

[KK00] Brad Karp and H. T. Kung. GPSR: Greedy perimeter stateless routing for wireless networks. In *Proceedings of the Sixth annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 2000)*, pages 243–254, August 2000.

[KLX+02] Jiejun Kong, Haiyun Luo, Kaixin Xu, Daniel Lihui Gu, Mario Gerla, and Songwu Lu. Adaptive security for multi-layer ad-hoc networks. *Special Issue of Wireless Communications and Mobile Computing*, August 2002.

[Knu97] Donald E. Knuth. *The art of computer programming*. Addison-Wesley, Reading, Massachusetts, USA, third edition, 1997.

[KS78] Leonard Kleinrock and John A. Silvester. Optimum transmission radii for packet radio networks or why six is a magic number. In *Proceedings of the IEEE National Telecommunications Conference*, pages 4.3.1–4.3.5, Birmingham, Alabama, USA, December 1978.

[KSU99] Evangelos Kranakis, Harvinder Singh, and Jorge Urrutia. Compass routing on geometric networks. In *Proceedings of the Eleventh Canadian Conference on Computational Geometry*, pages 51–54, Vancouver, Canada, August 1999.

[KTC01] Dongkyun Kim, C.-K. Toh, and Yanghee Choi. On supporting link asymmetry in mobile ad hoc networks. In *Proceedings of*

the IEEE Global Telecommunications Conference (GLOBECOM), pages 2798–2803, San Antonio, Texas, USA, November 2001.

[KV98] Young-Bae Ko and Nitin H. Vaidya. Location-aided routing (lar) in mobile ad hoc networks. In *Proceedings of the Forth annual ACM/IEEE International Conference on Mobile Computing and Networking*, pages 66–75, Dallas, Texas, USA, 1998.

[KVCP97] P. Krishna, NH. Vaidya, M. Chatterjee, and DK. Pradhan. A cluster-based approach for routing in dynamic networks. *ACM SIGCOMM Computer Communication Review*, 27(2):49–65, April 1997.

[KWZ02] Fabian Kuhn, Roger Wattenhofer, and Aaron Zollinger. Asymptotically optimal geometric mobile ad-hoc routing. In *Proceedings of the Sixth International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, pages 24–33, Atlanta, Georgia, USA, September 2002.

[KWZ03a] Fabian Kuhn, Roger Wattenhofer, and Aaron Zollinger. Ad-hoc networks beyond unit disk graphs. In *Proceedings of the 2003 Joint Workshop on Foundations of Mobile Computing (DIALM-POMC)*, pages 69–78, San Diego, CA, USA, 2003.

[KWZ03b] Fabian Kuhn, Roger Wattenhofer, and Aaron Zollinger. Worst-case optimal and average-case efficient geometric ad-hoc routing. In *Proceedings of the Fourth ACM International Symposium on Mobile ad hoc Networking & Computing*, pages 267–278, Annapolis, Maryland, USA, 2003. ACM Press.

[Lam77] Leslie Lamport. Proving the correctness of multiprocess programs. *IEEE Transactions on Software Engineering*, 3(2):125–143, March 1977.

[LBC+01] Jinyang Li, Charles Blake, Douglas S.J. De Couto, Hu Imm Lee, and Robert Morris. Capacity of ad hoc wireless networks. In *Proceedings of the Seventh annual International Conference on Mobile Computing and Networking (MobiCom)*, pages 61–69, 2001.

[LCWW03] Xiang-Yang Li, Gruia Calinescu, Peng-Jun Wan, and Yu Wang. Localized delaunay triangulation with application in ad hoc wireless networks. *IEEE Transactions on Parallel and Distributed Systems*, 14(10):1035–1047, October 2003.

[LH04] Ning Li and Jennifer C. Hou. Topology control in heterogeneous wireless networks: problems and solutions. In *Proceedings of the Twentythird Conference of the IEEE Communications Society (INFOCOM)*, Hong Kong, China, March 2004. The version in the conference proceedings has a technical error, and an updated version of the paper appears as a technical report at Department of Computer Science, University of Illinois at Urbana Champaign, Technical Report No. UIUCDCS-R-2004-2412, March 2004.

[LHB+01] Li Li, Joseph Y. Halpern, Paramvir Bahl, Yi-Min Wang, and Roger Wattenhofer. Analysis of a cone-based distributed topology control algorithm for wireless multi-hop networks. In *Proceedings of the Twentieth annual ACM Symposium on Principles of Distributed Computing (PODC)*, pages 264–273, Newport, Rhode Island, USA, August 2001.

[LHS05] Ning Li, Jennifer C. Hou, and Lui Sha. Design and analysis of an mst-based topology control algorithm. *IEEE Transactions on Wireless Communications*, 4(3):1195–1206, May 2005.

[LJC+00] Jinyang Li, John Jannotti, Douglas S.J. De Couto, David R. Karger, and Robert Morris. A scalable location service for geographic ad hoc routing. In *Proceedings of the Sixth annual International Conference on Mobile Computing and Networking (MobiCom)*, pages 120–130, Boston, Massachusetts, USA, 2000.

[LLM+02] Errol L. Lloyd, Rui Liu, Madhav V. Marathe, Ram Ramanathan, and S. S. Ravi. Algorithmic aspects of topology control problems for ad hoc networks. In *Proceedings of the Third ACM International Symposium on Mobile ad hoc Networking & Computing (MobiHoc)*, pages 123–134, Lausanne, Switzerland, 2002.

[LS03] Xu Lin and Ivan Stojmenovic. Location based localized alternate, disjoint and multi-path routing algorithms for wireless networks. *Journal of Parallel and Distributed Computing*, 63(1):22–32, 2003.

[Luk99] Tamás Lukovszki. *New Results on Geometric Spanners and Their Applications*. PhD thesis, University of Paderborn, 1999.

[LWW01] Xiang-Yang Li, Peng-Jun Wan, and Yu Wang. Power efficient and sparse spanner for wireless ad hoc networks. In *Proceedings of the IEEE International Conference on Computer Communications and Networks (ICCCN)*, Scottscale, Arizona, USA, October 2001.

[MC] Joseph Macker and Scott Corson. IETF mobile ad hoc networks. http://www.ietf.org/html.charters/manet-charter.html. IETF Mobile Ad Hoc Networks (MANET) Charter.

[MD02] Mahesh K. Marina and Samir R. Das. Routing performance in the presence of unidirectional links in multihop wireless networks. In *Proceedings of the Third ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, pages 12–23, June 2002.

[MGLA96] Shree Murthy and Jose Joaquin Garcia-Luna-Aceves. An efficient routing protocol for wireless networks. *Mobile Networks and Applications*, 1(2):183–197, October 1996.

[MM04] C. Siva Ram Murthy and B.S. Manoj. *Ad Hoc Wireless Networks: Architectures and Protocols*. Prentice Hall Communications Engeineering and Emerging Technologies Series. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2004.

[MS80] David W. Matula and Robert R. Sokal. Properties of gabriel graphs relevant to geographic variation research and the clustering of points in the plane. *Geographical Analysis*, 12(3):205–222, July 1980.

[MWH01] Martin Mauve, Jörg Widmer, and Hannes Hartenstein. A survey on position-based routing in mobile ad hoc networks. *IEEE Network Magazine*, 15(6):30–39, November 2001.

[NI97] Julio C. Navas and Tomasz Imielinski. Geocast – geographic addressing and routing. In *Proceedings of Third ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, pages 66–76, Budapest, Hungary, September 1997.

[NK84] Randolph Nelso and Leonard Kleinrock. The spatial capacity of a slotted aloha multihop packet radio network with capture. *IEEE Transactions on Communications*, 32(6):684–694, June 1984.

[NN01] Dragos Niculescu and Badri Nath. Ad hoc positioning system (APS). In *Proceedings of the IEEE Global Telecommunications Conference GLOBECOM*, pages 2926–2931, November 2001.

[NTCS99] Sze-Yao Ni, Yu-Chee Tseng, Yuh-Shyan Chen, and Jang-Ping Sheu. The broadcast storm problem in a mobile ad hoc network.

In *Proceedings of the Fifth annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, pages 151–162, Seattle, Washington, United States, 1999.

[PB94] Charles Perkins and Pravin Bhagwat. Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. In *Proceedings of the ACM SIGCOMM Conference on Communications Architectures, Protocols and Applications*, pages 234–244, August 1994.

[PC97] Vincent D. Park and M. Scott Corson. A highly adaptive distributed routing algorithm for mobile wireless networks. In *Proceedings of the Sixteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Driving the Information Revolution (INFOCOM)*, pages 1405–1413, April 1997.

[Pen99] Mathew D. Penrose. On k-connectivity for a geometric random graph. *Random Structures & Algorithms*, 15(2):145–164, September 1999.

[PH03] Panagiotis Papadimitratos and Zygmunt J. Haas. Secure link state routing for mobile ad hoc networks. In *Proceedings of the 2003 Symposium on Applications and the Internet Workshops (SAINT'03 Workshops)*, pages 379–383. IEEE Computer Society, January 2003.

[PHM00] Marc R. Pearlman, Zygmunt J. Haas, and Benjamin P. Manvell. Using multi-hop acknowledgements to discover and reliably communicate over unidirectional links in ad hoc networks. In *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC)*, pages 532–537, September 2000.

[PR99] Charles E. Perkins and Elizabeth M. Royer. Ad-hoc on-demand distance vector routing. In *Proceedings of the Second IEEE Workshop on Mobile Computer Systems and Applications*, pages 90–100, February 1999.

[Pra01] Ravi Prakash. A routing algorithm for wireless ad hoc networks with unidirectional links. *Wireless Networks*, 7(6):617–625, 2001.

[PS85] Franco P. Preparata and Michael Ian Shamos. *Computational Geometry - An Introduction. Texts and Monographs in Computer Sciences*. Springer Verlag, New York, Berlin, Heidelberg, Tokyo, 1985.

[Raj02] Rajmohan Rajaraman. Topology control and routing in ad hoc networks: A survey. *ACM SIGACT News*, 33(2):60–73, June 2002.

[Rap01] Theodore S. Rappaport. *Wireless Communications: Principles and Practice*. Prentice Hall Communications Engeineering and Emerging Technologies Series. Prentice Hall PTR, Upper Saddle River, NJ, USA, second edition, 2001.

[RCM02] Venugopalan Ramasubramanian, Ranveer Chandra, and Daniel Mosse. Providing a bidirectional abstraction for unidirectional ad hoc networks. In *Proceedings of the Twentyfirst annual Joint Conference of the IEEE Computer and Communications Societies(INFOCOM)*, pages 1258–1267, 2002.

[RM99] Volkan Rodoplu and Teresa H. Meng. Minimum energy mobile wireless networks. *IEEE Journal on Selected Areas in Communications*, 17(8):1333–1344, August 1999.

[RRH00] Ram Ramanathan and Regina Rosales-Hain. Topology control of multihop wireless networks using transmit power adjustment. In *Proceedings of the IEEE Conference on Computer Communications INFOCOM*, pages 404–413, Tel Aviv, Israel, March 2000.

[RS91] Jim Ruppert and Raimund Seidel. Approximating the d-dimensional complete euclidean graph. In *Proceedings of the Third Canadian Conference on Computational Geometry (CCCG)*, pages 207–210, 1991.

[RT99] Elizabeth M. Royer and Chai-Keong Toh. A review of current routing protocols for ad-hoc mobile wireless networks. *IEEE Personal Communications Magazine*, 6(2):46–55, April 1999.

[San05] Paolo Santi. Topology control in wireless ad hoc and sensor networks. *ACM Computing Surveys*, 37(2):164–194, 2005.

[SD95] Peter Su and Robert L. Scot Drysdale. A comparison of sequential delaunay triangulation algorithms. In *Proceedings of the Eleventh annual ACM Symposium on Computational Geometry*, pages 61–70, Vancouver, British Columbia, Canada, June 1995.

[SL01] Ivan Stojmenovic and Xu Lin. Loop-free hybrid single-path/flooding routing algorithms with guaranteed delivery for wireless networks. *IEEE Transactions on Parallel and Distributed Systems*, 12(10):1023–1032, October 2001.

[SRL02] Chris Savarese, Jan Rabaey, and Koen Langendoen. Robust positioning algorithms for distributed ad-hoc wireless sensor networks. In *Proceedings of the General Track: 2002 USENIX Annual Technical Conference*, pages 317–327, June 2002.

[Sto99] Ivan Stojmenovic. Voronoi diagram and convex hull based geocasting and routing in wireless networks. Technical Report TR-99-11, SITE, University of Ottawa, December 1999.

[Str04a] Hannes Stratil. Design of a voronoi-aided routing (VAR) protocol for wireless sensor networks. Research Report 15/2004, Technische Universität Wien, Institut für Technische Informatik, Treitlstraße 3, A-1040 Vienna, Austria, 2004.

[Str04b] Hannes Stratil. An efficient implementation of the greedy forwarding strategy. In *Proceedings of the Workshop on Sensor Networks*, pages 365 – 369, Ulm, Germany, September 2004.

[Str05a] Hannes Stratil. Distributed construction of an underlay in wireless networks. In *Proceedings of the Second European Workshop on Wireless Sensor Networks*, pages 176–187, Istanbul, Turkey, January 2005.

[Str05b] Hannes Stratil. Fault tolerant topology control with unreliable failure detectors. In *Proceedings of the Seventeenth IASTED International Conference on Parallel and Distributed Computing and Systems*, Phoenix, Arizona, USA, November 2005.

[Sup83] Kenneth J. Supowit. The relative neighborhood graph, with an application to minimum spanning trees. *Journal of the ACM*, 30(3):428–448, 1983.

[SV99] Ivan Stojmenovic and Bosko Vukojevic. A routing strategy and quorum based location update scheme for ad hoc wireless networks. Technical Report TR-99-09, SITE, University of Ottawa, September 1999.

[SW04] Ivan Stojmenovic and Jie Wu. Broadcasting and activity-scheduling in ad hoc networks. In Stefano Basagni, Marco Contia, Silvia Giordano, and Ivan Stojmenovic, editors, *Ad Hoc Networking*, pages 205–229. Wiley–IEEE Press, 2004.

[SWLF04] Wen-Zhan Song, Yu Wang, Xiang-Yang Li, and Ophir Frieder. Localized algorithms for energy efficient topology in wireless ad hoc

networks. In *Proceedings of the Fifth ACM International Symposium on Mobile ad hoc Networking and Computing*, pages 98–108, Roppongi Hills, Tokyo, Japan, 2004.

[SZHK04] Karim Seada, Marco Zuniga, Ahmed Helmy, and Bhaskar Krishnamachari. Energy efficient forwarding strategies for geographic routing in wireless sensor networks. In *Proceedings of the ACM Conference on Embedded Networked Sensor Systems (SenSys)*, Baltimore, Maryland, USA, November 2004.

[TK84] Hideaki Takagi and Leonard Kleinrock. Optimal transmission ranges for randomly distributed packet radio terminals. *IEEE Transactions on Communications*, 32(3):246–257, March 1984.

[Toh97] Chai-Keong Toh. Associativity-based routing for ad hoc mobile networks. *Wireless Personal Communications: An International Journal*, 4(2):103–139, March 1997.

[Tou80] Godfried T. Toussaint. The relative neighborhood graph of a finite planar set. *Pattern Recognition*, 12(4):261–268, 1980.

[Urq83] Roderick B. Urquhart. Some properties of the planar euclidean relative neighbourhood graph. *Pattern Recognition Letters*, 1(5):317–322, 1983.

[Vor07] Georgii Feodosevich Voronoi. Nouvelles applications des paramètres continus à la thèorie des formes quadratiques. premier mémoire: Sur quelques proprietés de formes quadratiques positives parfaites. *Journal für die Reine und Angewandte Mathematik*, 133:97–178, 1907. in French.

[Vor08] Georgii Feodosevich Voronoi. Nouvelles applications des paramètres continus à la théorie des formes quadratiques. deuxième mémoire: Recherches sur les parallélloèdres primitifs. *Journal für die Reine und Angewandte Mathematik*, 134:198–287, 1908. in French.

[Vor09] Georgii Feodosevich Voronoi. Nouvelles applications des paramètres continus à la théorie des formes quadratiques. deuxième mémoire: Recherches sur les parallélloèdres primitifs. *Journal für die Reine und Angewandte Mathematik*, 136:67–181, 1909. in French.

[Wat81] David F. Watson. Computing the n-dimensional delaunay tesselation with application to voronoi polytopes. *The Computer Journal*, 24(2):167–172, 1981.

[WLBW01] Roger Wattenhofer, Li Li, Paramvir Bahl, and Yi-Min Wang. Distributed topology control for wireless multihop ad-hoc networks. In *Proceedings of the Twentieth annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, pages 1388–1397, Anchorage, Alaska, USA, April 2001.

[WLF03] Yu Wang, Xiang-Yang Li, and Ophir Frieder. Distributed spanners with bounded degree for wireless ad hoc networks. *International Journal of Foundations of Computer Science*, 14(2):183–200, April 2003.

[WLMN+03] WeiZhao Wang, Xiang-Yang Li, Kousha Moaveni-Nejad, Yu Wang, and Wen-Zhan Song. The spanning ratios of beta-skeletons. In *Proceedings of the Canadian Computational Conference on Geometry (CCCG)*, August 2003.

[WZ04] Roger Wattenhofer and Aaron Zollinger. XTC: A practical topology control algorithm for ad-hoc networks. In *Proceedings of the Forth International IEEE Workshop on Algorithms for Wireless, Mobile, Ad Hoc and Sensor Networks (WMAN)*, Santa Fe, New Mexico, USA, April 2004.

[Yao82] Andrew Chi-Chih Yao. On constructing minimum spanning trees in k-dimensional spaces and related problems. *SIAM Journal on Computing*, 11(4):721–736, November 1982.

Curriculum Vitae

Biographical Information:

Name:	DI Hannes Stratil
Date of Birth:	May 12, 1976
Place of Birth:	Bruck an der Mur, Styria, Austria
Citizenship:	Austrian
Address:	Grazerstrae 85a, 8605 Kapfenberg, Austria

Education History:

since Mar. 2005	DOC scholarship founded by the Austrian Academy of Science
since Mar. 2002	Doctorate study in Engineering Science Vienna University of Technology
March 12, 2002	Master degree in Computer Science with the thesis: *Topology Management and Routing in Wireless Networks – An Overview*
Oct. 2001 – Dec. 2003	Research Fellow, Project: W2F, founded by the Austrian START-Programme under the aegis of Prof. Dr. Ulrich Schmid
Oct. 2000 – Apr. 2001	Study Abroad University of Lancaster, UK Research Fellow, Project: DESARTE, founded by the European Commission
1996 – 2002	Diploma study of Computer Science Vienna University of Technology University of Lancaster, UK
Oct. 1995 – May 1996	Military service in the Austrian Army Landwehrkaserne St.Michael
June 1995	High School Certificate
1990 – 1995	Technical High School for Electrical Engineering Kapfenberg, Austria
1982 – 1990	Elementary and Secondary School