Urban VANET Performance Optimization

by

Xiang Yu

A dissertation submitted in fulfillment of the

Requirements for the degree of Doctor of Philosophy

University of Warwick, School of Engineering

20/12/2013

Foreword

Table of Contents

Acknowledgements

I would like to take this opportunity to thank my supervisors, Dr. Mark Leeson and Prof. Evor Hines, for their academic support, supervision during my Ph.D stage, and motivation whenever I was depressed. Without their contributions this work will never come to a reality.

I would also like to thank my colleagues working with me in A405a lab, your academic excellence and inspirational spirit always encouraged me. The group discussion and your valuable feedback to my project benefit me a lot. Wish you all have a splendid future.

Special thanks here to my family: my dad and mum who sponsored me in this project, my wife Sue always supported me when I was in difficulties and depression. Without your encouragement I will never finish this thesis.

Last but not least, thanks to all academic staff who discussed with me about my projects: Dr. Weisi Guo, Prof. Nigel Stocks, Dr. Christos Mias … thank you for your valuable suggestions and wish we can work together in future.

Declaration

The work described in this thesis was conducted by the author, except where stated otherwise, in the School of Engineering, University of Warwick between the dates of March 2010 and December 2013. No part of this work has been previously submitted to the University of Warwick or any other academic institution for admission to a higher degree. All publications to date arising from this thesis are listed in the next section.

Abstract

Urban VANET (Vehicular Ad hoc NETworks) performance optimization concerns the improvement of wireless signal quality between two arbitrary selected nodes moving within along city streets. It includes three procedures: VANET architecture modeling; wireless signal simulation; and signal quality optimization techniques. The first procedure converts real-world map data into a network graph according to the requirement of the optimization algorithm. The second step analyzes a communication route between two network nodes and calculates received signal quality with the information provided by the network model. The final operation optimizes the signal quality to an expected level by choosing appropriate communication route between two wireless nodes.

In this thesis, three optimization techniques are presented: EP (Evolutionary Programming), SG (Stochastic Geometry) and SW (Small World). EP is a widely applied optimization strategy based on Darwin's natural selection and evolution theory. It is effective with an enormous number of data support, and it can provide detailed route information. However, it requires enough time to evolve to an optimal solution. SG is a statistical tool to analyze points' distribution within a multi-dimensional space, and it was recently applied on wireless network analysis. Given the distribution characteristics of an urban area, SG can calculate average data loss rate of a communication route. However, it cannot provide detailed route information. SW is a widely accepted model to represent people's relationship in social networks, and it can be used

in VANET analysis. SW provides a simplified network architecture compared with EP an SG. However, it requests additional long-range communication equipment and consumes more energy.

The thesis is divided into three parts. Chapter 1 introduces the history of VANET and its architecture (in this research, it is a combination of Ad hoc network and WSN (Wireless Sensor Network). Chapter 2 and 3 presents literature review of EP and SG. Chapter 4, 5, and 6 discusses how to implement EP, SG and SW on Boston VANET. At the end of each chapter, a conclusion is presented and a discussion on the author's contribution is given.

List of Figures

List of tables

CHAPTER 1

VEHICULAR AD HOC NETWORKS

In this chapter a general overview is presented of VANETs (Vehicular Ad hoc NETworks), the domain in which this research was carried out. An introduction of the origin of the VANET—wireless network (history, technical breakthrough, current status) is firstly given. Then follows a brief review of two widely studied models in this field: ad hoc networks, and wireless sensor networks, especially focusing on their network routing protocols. Since the VANET model that has been applied in this research consists of ad hoc network node pairs and wireless sensor routers, the design of a routing protocol needs to consider both of the networks' characteristics. Finally a brief of key conferences and organizations studying VANET is presented.

1.1 INTRODUCTION TO WIRELESS NETWORKS [1.1]

The use of wireless signals (signals whose frequencies are within the range 9 kHz to 300 GHz) in telecommunications can be dated back to the 1940s, when in the Second World War radio waves were applied for military purposes. However, this technology was sealed within laboratories until long after the war. At the beginning of the 1950s, Bell Labs in the United States deployed the first generation of wireless networks called the AMPS [1.2] (Advanced Mobile Phone Service) network. Then it was standardized and commercialized in North America in 1982. An AMPS network operated at a central frequency of 850 MHz, and it could support 21 control channels, as well as 395 voice channels. These channels owned 30 kHz uplink or downlink bandwidths, transferring analog signals. From then on, the evolution of wireless networks has never ceased.

Until this very day, it has experienced four generations with a family of many different standardized versions deployed in different areas. The latest wireless network, branded as LTE [1.3] (Long Term Evolution) operates on various frequency bands (700, 800, 1900 and 1700/2100 MHz in North America; 2500 MHz in South America; 800, 900, 1800, 2600 MHz in Europe; 1800 and 2600 MHz in Asia; and 1800 MHz in Australia and New Zealand), supporting peak download rates of 300 Mbps and upload rates of 75.4 Mbps. Its channels are transferring digital signals coded with vast many multimedia standards (for example: JPEG for image compression, MPEG 2000 for video compression, and AAC-ELD, Advanced Audio Coding for Enhanced Latency Delay). In addition, current wireless networks are not only interfacing with the PSTN (Public Switched Telephone Network), but also interoperating with the Internet, providing IP-based services [1.4]. From its physical infrastructure to virtual routing protocols, wireless networking is evolving more as an all IP network, in common with the Internet. Many Internet based techniques can be seamlessly applied on wireless networks, with only minor modifications to physical settings.

In the past decade, wireless networks have been studied by many researchers due to their widespread application in daily life. However, much of the work has been carried out in the free-space environment and assumed that signal propagation follows a simple line of sight (LOS) transmission model [1.5], [1.6]. There was thus a dearth of research were executed in the real urban area environment, where wireless signals need to transmit through buildings and across concrete walls before reaching their destinations. This situation lasted until the last decade, when a transportation based system, the vehicular ad-hoc Network (VANET) was introduced and investigated in several industrial projects [1.7], [1.8], [1.9]. As described in the subsequent reports and surveys, a VANET is a mobile ad-hoc network (MANET), where vehicles act as nodes which are restricted to move along city streets[1.10], [1.11]. The VANET is clearly a more realistic modeling of urban ad-hoc networks.

This research explores the possibility of communications on one and more pairs of VANET nodes through a sensor network. The signals originate and terminate both on VANET nodes, which are restricted in an urban access area (streets). However, these signals are forwarded through many sensor routers before reaching their destinations. Many types of routing protocols exist in MANETs and wireless sensor networks (WSNs), and here these are reviewed before the creation of a new strategy and its optimization. In section 1.2 and 1.3, these two types of wireless networks and their routing protocols are introduced. In section 1.4, an introduction to current mainstream international conferences and journals is presented.

1.2 INTRODUCTION TO AD HOC NETWORKS [1.12]

Ad-hoc, itself a Latin phrase meaning 'to this'. When used as a scientific term, it generally specifies an object whose nature is flexible or unpredictable. As a member of the wireless network family, ad-hoc networks or MANETs, were firstly introduced in the 1970s. The ALOHA network [1.13] which was carried at University of Hawaii and sponsored by DARPA (Defense Advanced Research Project Agency) was the first generation of today's ad-hoc networks. Current ad-hoc networks refer themselves to a global specification series of IEEE

802.11 [1.14], standardized and released at 1997 by the IET. A brief review has been given to ad-hoc routing protocols which are directly associated with the research carried in this project. Chapter 4 further analyzes typical routing protocols in ad-hoc networks.

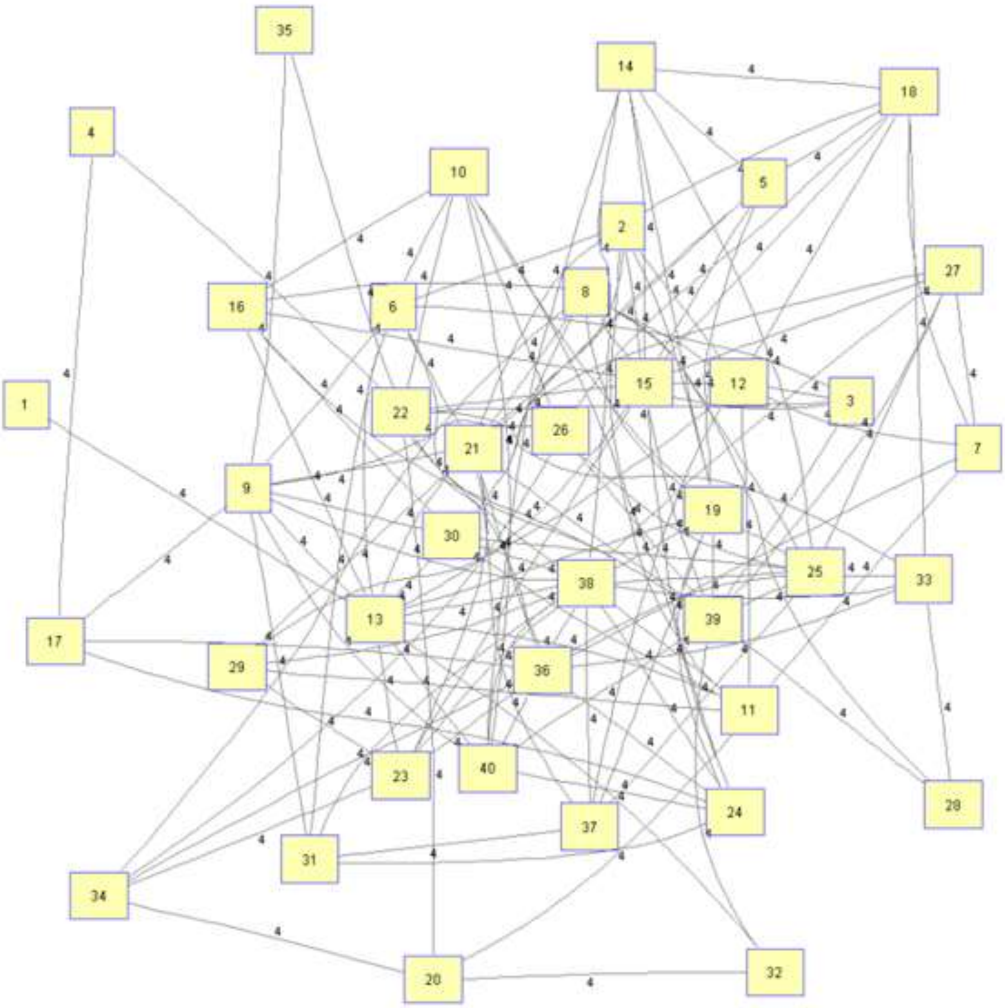


Fig. 1.1 A 40-node ad-hoc network frame

(created by Matlab Gaussian distribution function)

In Fig. 1.1, a 40-node Gaussian distribution network was simulated in Matlab to represent an ad-hoc network frame in reality. One method to simulate such a network is to use the Random Way Point model, which can be simulated using a Matlab reference design [1.32].

According to [1.15], existing ad-hoc network routing protocols can be broadly classified into as two types:

1. Table driven
2. Source initiated

In table-driven routing protocols, ad-hoc network information is stored in routing tables maintained separately by its node members. This information may be their distances to a sourcing node, a cost defined by a specific protocol, or anything useful in a network routing algorithm. Any changes to this information will be broadcast within the whole network, and routing tables will be updated accordingly. A routing enquiry from one node to another is transformed into a search of its routing table information. The protocol will firstly seek for a direct match in its routing table, and no entry is found, seek for an intermediate node (called a hop) to receive this enquiry. The table-driven routing technique was firstly used on the Internet, where equipment (routers) is specially designed to maintain a table on all IP requests flowing through it and direct different requests to their destinations according to its table recordings. Later on it was successfully applied on wireless ad-hoc networks. However, as the fast-changing nature of an ad-hoc network, routing information could expire in a shorter period, making it very difficult to be precise.

Currently wide-accepted table-driven routing protocols in ad hoc networks are:

a. Destination-Sequenced Distance-Vector Routing [1.16]

b. Cluster head Gateway Switch Routing [1.17]

c. Wireless Routing Protocol [1.18]

Differently to the table-driven routing technique, source-initiated on-demand routing is a reactive process only responding to source node requests. When a node in wireless network requires a communication to another node, it initiates a connection request to the protocol. The protocol will then discover a route for it and maintain this route until the communication is over.

a) Ad hoc On-Demand Distance Vector Routing [1.19]

b) Dynamic Source Routing [1.20]

c) Temporarily Ordered Routing Algorithm [1.21]

d) Associability-Based Routing [1.22]

e) Signal Stability Routing [1.23]

These routing protocols share common characteristics, as are shown in table 1.1 [1.15]: they all target on flat network architecture; they are loop-free so they can avoid dead lock in a routing search; most of them are unicast, and store their routing information in routing tables; many of them are using the shortest path algorithm to look for the next hop. In conclusion, these routing protocols are very similar, so AODV is chosen to represent this source-initiating protocol family.

Table 1.1 Characteristics of source initiated ad hoc routing protocols [1.15]

Performance parameters	AODV	DSR	TORA	ABR	SSR
Time complexity (initial)	O(2d)	O(2d)	O(2d)	O(d+z)	O(d+z)
Time complexity (recover)	O(2d)	O(2d)	O(2d)	O(l+z)	O(l+z)
Communication (initial)	O(2N)	O(2N)	O(N+y)	O(N+y)	O(N+y)
Communication (failure)	O(2N)	O(2N)	O(2x)	O(x+y)	O(x+y)
Routing philosophy	Flat	Flat	Flat	Flat	Flat
Multicast capability	Yes	No	No	No	No
Beaconing requests	No	No	No	Yes	Yes
Multiple routing capabilities	No	Yes	Yes	No	No
Routing maintained in	Route table	Route table	Table	Table	Table
Route cache/expiration timers	Yes	No	No	No	No
Route reconfiguration	Erase route; notify source	notify source	Link reversal	broadcast query	notify source

Abbreviations:

l = diameter of the network; y = total number of nodes forming the directed path where the REPLY packet transits; z = Diameter of the directed path where the REPLY packet transits

Current challenges for ad-hoc network routing protocols include:

- Multicast capabilities
- QoS in highly dynamic scenario
- Network security
- Interoperability with other types of networks (INTENET, Sensor networks, etc)

To develop and maintain a multicast routing structure is very difficult for an ad-hoc network because the multicast tree is no longer static. A dynamic routing strategy was mentioned in [1.24] to cope with dynamic multicast decisions (for example, routing reestablishment strategies when nodes join or leave the tree). How to maintain a stable, primary set of active connections under a highly dynamic environment needs to be considered by researchers.

Another important issue to solve is the Quality of Service (QoS) to improve end user experience. Different level of customer needs brings different QoS demands of each active connection, and when a customer needs upgrades, the routing protocol needs to allocate (reserve) enough network resources to satisfy it. This kind of reservation may lead to a level of network redundancy, and a well-designed routing protocol shall locate a balance point between redundancy and efficiency.

Finally, more and more researchers now focus on interoperability between ad-hoc networks and other types of networks. In this research, the possibility of transmitting ad hoc signals through a sensor network is also explored. As is shown in later sections, pre-deployed sensor networks can reduce point-to-point communication bit error rates with lower energy consumption. Although it may bring higher delay due to an enormous number of hops, it is still a candidate for backup connections. A review of WSNs is given in the next section.

1.3 INTRODUCTION TO WIRELESS SENSOR NETWORKS

In contrast to the ad hoc network infrastructure, a WSN contains many more nodes (hundreds, often thousands), but these sensors are lacking mobility, with limited power supply, and can only perform very few communication tasks. In [18] the routing protocols are roughly classified into three types:

i. Flat
ii. Hierarchical
iii. Location-based

Also these routing techniques can be divided into five categories according to the routing information they use:

a. Multipath-based
b. Query-based
c. Negotiation-based
d. QOS (Quality of Service)-based
e. Coherent-based

Many of routing protocols applied on WSNs are designed to save energy, which is a major concern in this network supplied with limited power resources. However, the third category: location based routing protocols, is the key part within this research. So, a closer review will be given to this area.

A mandate for a WSN to operate a location based routing protocol is to have its sensor nodes GPS (Global Positioning System) assisted. Current protocols include:

a. GEAR (Geographic and Energy Aware Routing) [1.26]
b. GEDIR (Geographic Distance Routing) [1.27]
c. GOAFR (The Greedy Other Adaptive Face Routing) [1.28]
d. SPAN [1.29]

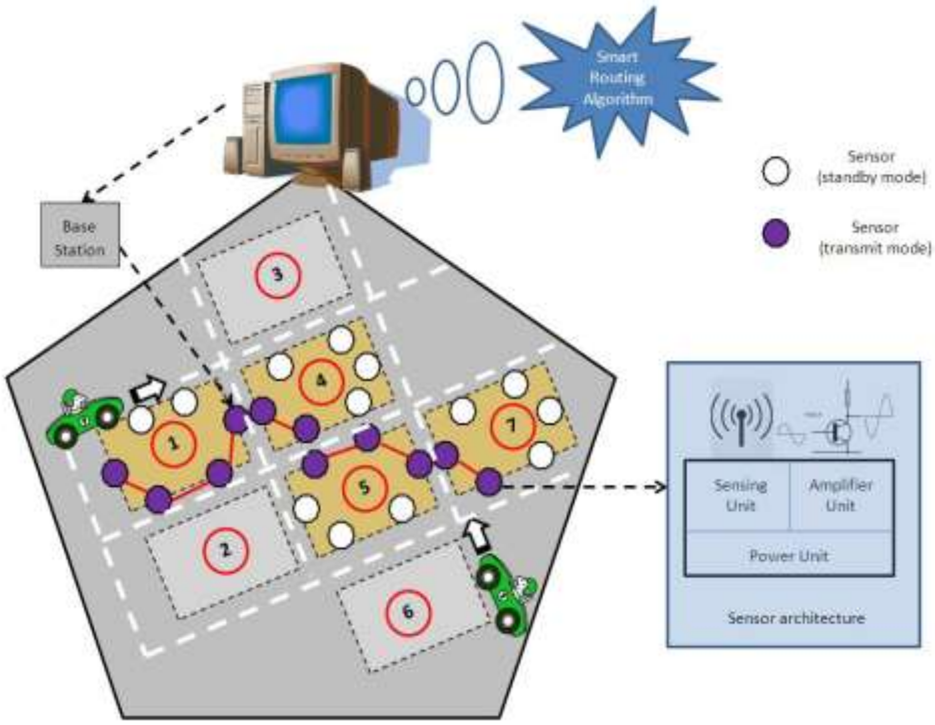


Fig. 1.2 A sample WSN diagram in an urban area

A sample wireless repeater network diagram is shown in Fig. 1.2, where wireless repeaters are deployed along city streets. An active communication link has been established between two vehicles. All wireless repeaters on this link are activated, and the rest are in standby mode. This diagram is used in the simulation in Chapter 4.

The technical challenges of routing protocol design in WSN remain in 3 aspects: energy saving, multipath routing, and data-processing capabilities. Until now extensive efforts have been put into design of energy efficient routing solutions, because sensor nodes can only have limited battery power. However, in the future, if these nodes can utilize natural power (solar or wind energy), the design of routing protocols can foresee a big improvement. Also, because sensor nodes lack of data processing and data storage units, they cannot compute and memorize complex data structures. A common case is to store these routing information into a sink unit (or base station), which manages a group of sensors adjacent to it. Distributed routing strategy is not suitable for today's WSNs, but they will be the future solution.

1.4 CONFERENCES AND ORGANIZATIONS

On 2nd of June 2013, the 5th International Symposium on Wireless Vehicular Communications (WIVEC) was held in Dresden, Germany [1.30]. In its 'calling for papers' section, the following areas were shown great interest:

1. Antenna design, physical layer and propagation models
2. Radio resource management and interference management
3. Vehicular Networking Architectures
4. **Networking protocols (including ad-hoc communication, routing, geo-casting, etc.) and their evaluation**
5. **Simulation of Vehicular Communications Systems**
6. **QoS and cross-layer optimization design**
7. Security, liability and privacy
8. Interworking with sensor network technologies

9. In-car electronics and embedded integration of wireless vehicular communications
10. Roadside infrastructure
11. Mobility management, mobility and vehicle traffic models
12. **Digital maps** and location technologies
13. Human Machine interface
14. Applications (eCall, eTolling, traffic information systems, safety applications, wireless diagnosis, etc.)
15. Standards development, business models, policies
16. Assessment of impact on transport efficiency and safety

The Panel of this conference addressed the importance of VANET as below:

"VANETs—from research to initial deployment—what comes next? Vehicular ad hoc networks (VANETs) have been subject to research for more than a decade now. Standardization development organizations in Europe and North America have already released an initial set of VANET standards. Field operation tests are currently being carried out. Industry consortia are now planning deployment of the first basic system. Based on different viewpoints and experiences from recent research studies, field trials and ongoing standardization process, the panel will discuss the remaining and next steps towards full deployment and aims to conclude on the coming key research questions. The panel with prominent speakers from academia as well as industry covers radio, communication protocols, applications and security."

IEEE Transactions on Vehicular Technology (TVT) [1.31]

Vehicular Technology (TVT) is an important section in the IEEE Transactions series, mainly targeting on the following areas:

1. Mobile communications
2. Transportation systems
3. Vehicular electronics

In 2010, this journal received 486,646 downloads in the IEEE Xplore usage, ranking 21st among 288 IEEE journals. Its impact factor has been increasing steadily since 2004, as shown in table 1.2.

Table 1.2. Impact Factor of TVT [1.31]

Year	2004	2005	2006	2007	2008	2009	2010
IF	0.611	0.860	1.071	1.191	1.308	1.488	1.490

Furthermore, IEEE defined a series of industrial standards on vehicular ad hoc networks, such as:

- 1609.3-2010 Wireless Access in Vehicular Environments (WAVE): Networking Services
- 1609.4-2010 Wireless Access in Vehicular Environments (WAVE): Multi-channel Operations

- 1609.11-2010 Wireless Access in Vehicular Environments (WAVE): Over-the-Air Electronic Payment Data Exchange Protocol for Intelligent Transportation Systems (ITS)

As can be seen from the facts above, the research society of VANETs is still active and productive. The major contributions it provides are not only academic, but also targeting on daily life. Extensive research interests have been put into interactions between personal devices to landmarks (vehicles, skyscrapers, etc). It also helps in standardization of VANETs.

1.5 CONCLUSIONS

This has firstly reviewed the emergence and development of wireless networks. Then a brief introduction was given to a special wireless network: the VANET. Since 2000s, enormous research efforts have been put on it because of its importance to urban life. In this research, the primary study is of a type of VANET consisting of ad hoc networks and WSNs, so a literature review was given for both of them. Since routing protocol design is the main concern here, the current status of routing protocols in each type of network was examined. Finally, the major research bodies (international conferences and IEEE societies) have been listed to keep track of newest information and research focus.

From this chapter it may be concluded that a novel and reliable VANET routing system requires consideration of the network characteristics. An ad hoc network is too random to describe a VANET, while a WSN is too rigid to track vehicle movement. A novel idea is to combine an ad hoc network with a WSN, and this type of hybrid network will be sufficient to describe the nature of VANET. From the review it is not possible to find one single routing protocol which suits this hybrid network, so one has been developed in this thesis from a prototype mentioned in section 2.4. This network model is then optimized by evolutionary programming technique, which is extensively described in chapter 4. Also a second network architecture, based on stochastic geometry technique has been presented, in Chapter 5. Finally in chapter 6 a third network simulation model: small world architecture is presented to solve VANET routing problem. Comparison between these techniques implies that EP's performance exceeds the other techniques with detailed positioning information of mobile nodes available. This requires mobile nodes to have Global Positioning System (GPS) capability. Stochastic Geometry is more advantageous in predicting data loss rates of a certain link within an urban area with a constant node density. It provides feedback of average signal loss rate, which is beneficial for optimization algorithms. Small World technique requests additional network equipment and it changes network topology as well. However it simplifies the network architecture by adding long range communication links to the system. So it reduces the average hop count needed to establish a random communication, and improves the signal quality of it. To summarize, EP provides a cost effective solution to optimize signal quality of VANET, SG is an alternative statistic model to predict data loss rate and SW can be considered when long range communication equipment is in presence.

REFERENCES

1.1 C. Makaya and S. Pierre, Emerging wireless networks: concepts, techniques, and applications. Book ISBN: 9781439821367. CRC Press, 2012.

1.2 W. R. Young, AMPS Introduction, Background, and Objectives, Bell Systems Technical Journal, vol. 58, 1, pp. 1-14, Jan. 1979.

1.3 G. de la Roche, A. A. Glazunov, B. Allen, LTE-advanced and next generation wireless networks: channel modeling and propogation. Book ISBN: 9781119976707. Wiley, 2013.

1.4 C. Zhu and Y.N. Li, Advanced video communications over wireless networks, Book ISBN: 9781439880005, CRC Press, 2013.

1.5 D. B. Green, A. S. Obaidat, An accurate line of sight propagation performance model for ad-hoc 802.11 wireless LAN (WLAN) devices, Communications, 2002. ICC 2002. IEEE Conf. Vol. 5. 2002.

1.6 J. C. Liberti, T. S. Rappaport, A geometrically based model for line-of-sight multipath radio channels. Vehicular Technology Conference, Mobile Technology for the Human Race, IEEE 46th vol. 2, 1996.

1.7 D.A. Magder, P. Bosch, T. Klein, P. Polakos, L. Samuel, H.Viswanathan, 911-NOW: a network on wheels for emergency response and recovery operations, Bell Labs Technical Journal, vol. 11, no. 4, pp 113-133, 2007.

1.8 W. Enkelmann, FleetNet-applications for inter-vehicle communication, Proceedings of the IEEE Intelligent Vehicles Symposium, pp. 162-167, 2003.

1.9 D. Reichardt, M. Miglietta, L. Moretti, P. Morsink, W. Schulz, CarTALK2000: safe and comfortable driving based upon inter-vehicle communication. Intelligent Vehicles Symposium, 2002. Proceedings. IEEE, pp 545-550, 2002.

1.10 F. Li and W. Yu, Routing in vehicular ad hoc networks: A survey. IEEE Vehicular Technology Magazine, vol. 2, no. 2, pp. 12-22, 2007.

1.11 Y. Toor, P. Muhlethaler, and A. Laouiti, Vehicle Ad Hoc networks: applications and related technical issues, IEEE Communications Surveys and Tutorials, vol. 10, no. 3, pp. 74-88, 2008.

1.12 J. Loo, J. L. Mauri and H. Ortiz, Mobile ad hoc networks: current status and future trends, Book ISBN: 9781439856505, CRC Press, 2012.

1.13 F. F. Kuo, The ALOHA system, ACM Computer Communication Review: vol. 25, 1995.

1.14 T. Cooklev, Wireless communication standards: a study of IEEE 802.11, 802.15 and 802.16. Book ISBN: 9781118098837. IEEE Press, 2004.

1.15 E. M. Royer, C.K. Toh, A review of current routing protocols for Ad Hoc mobile Wireless Networks, pp 46-55, IEEE Personal Communications, April 1999.

1.16 C. E. Perkins and P. Bhagwat, Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers, pp 234-244, Computer Communications, October 1994.

1.17 C. C. Chiang, Routing in clustered multi-hop, mobile wireless networks with fading channel, Proceedings of IEEE SICON, April 1997.

1.18 S. Murthy and J. J. Garcia-Luca-Aceves, 'An Efficient Routing Protocol for Wireless Networks', ACM Mobile Networks and App. J., Special Issue on Routing in Mobile Communication Networks, pp. 183-97, 1996.

1.19 C. E. Perkins and E. M. Royer, 'Ad-hoc On-Demand Distance Vector Routing,' Proc. 2nd IEEE Workshop Mobile Comp. Sys. And Apps. , pp. 90-100, 1999.

1.20 D. B. Johnson and D. A. Maltz, 'Dynamic Source Routing Protocol for Mobile Ad Hoc Networks,' Mobile Computing, pp. 153-81, 1996.

1.21 V. D. Park and M. S. Corson, 'A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks,' Proc. Infocom'97, Apr. 1997.

1.22 C-K. Toh, 'A Novel Distributed Routing Protocol To Support Ad-Hoc Mobile Computing,' Proc. 1996 IEEE 15th Annual Int'l. Conf., pp. 480-96, 1996.

1.23 R. Dude et al., 'Signal Stability based Adaptive Routing (SSA) for Ad-Hoc Mobile Networks,' IEEE Pers. Communications, pp. 36-45, 1997.

1.24 M. Gerla, C. –C. Chiang, and L. Zhang, 'Tree Multicast Strategies in Mobile, Multihop Wireless Networks,', ACM/Baltzer Mobile Networks and Apps. J., 1998.

1.25 J. N. Al-karaki, Ahmed E. Kamal, 'Routing techniques in wireless sensor networks: a survey,' Wireless Communications, IEEE, Dec. 2004.

1.26 Y. Yu, D. Estrin, and R. Govindan, 'Geographical and Energy-Aware Routing: A Recursive Data Dissemination Protocol for Wireless Sensor Networks,' UCLA Comp. Sci. Dept. tech. rep., UCLA-CSD TR-010023, May 2001.

1.27 I. Stojmenovic and X. Lin, 'GEDIR: Loop-Free Location Based Routing in Wireless Networks,' Int'l Conf. Parallel and Distrib. Comp. and Sys., Boston, MA, Nov. 3-6, 1999.

1.28 F. Kuhn, R. Wattenhofer and A. Zollinger, 'Worst-Case Optimal and Average-Case Efficient Geometric Ad Hoc Routing,' Proc. 4th ACM Int'l. Conf. Mobile Comp. and Net., pp 267-78, 2003.

1.29 B. Chen et al., 'SPAN: an Energy-efficient Coordination Algorithm for Topology Maintenance in Ad Hoc Wireless Networks,' Wireless Networks, vol. 8, no. 5, pp. 481-94, 2002.

1.30 5th International Symposium on Wireless Vehicular Communications, Dresden Germany, http://www.ieeevtc.org/wivec2013/index.php. Accessed 10th November 2013.

1.31 IEEE Vehicular Technology Society, http://www.vtsociety.org/ . Accessed 10th November 2013.

1.32 Random Waypoint Model Matlab reference page:

http://www.mathworks.com/matlabcentral/fileexchange/30939-random-waypoint-mobility-model . Accessed 25th August 2012.

CHAPTER 2

EVOLUTIONARY PROGRAMMING

In this chapter an overview of evolutionary programming (EP) is given, since this is used as a major optimization tool in this research. EP assists the algorithms designed evolve from a population of routing results to reach a near-optimum objective. EP has been proved in scientific research as a reliable optimization tool. A literature review is firstly presented in section 2.1, followed by the introduction of a series of recently developed EP variants. An emphasis will be placed on multiple objective optimization approaches in section 2.2, which play a critical part in the research analysis. Finally, a specific example in the area of Dense Wavelength Division Multiplexing (DWDM) networking is reviewed in section 2.3 to illustrate EP in action.

2.1 HISTORY OF EVOLUTIONARY PROGRAMMING [2.1]

The term *optimization* comes from an idea to determine 'a best' solution to a certain mathematical problem. Such problems occur in many fields, such as economics, management, physics and engineering. For example, in the construction of buildings it is desirable to find an optimal solution to satisfy customer requirements with minimum material cost [2.2]. Furthermore, in numerical analysis such as data fitting, the need to find an equation with an optimal curve shape often arises. The solutions play a critical part in daily life, and the process to discover these solutions is called optimization.

In recent times, the true beginning of the exploration of the solution of optimization problems came from late 1950s, when an important method called *linear programming* was introduced [2.3]. The idea was to create a series of equations of a variable *x*, with an order of *n,* from practical conditions. Then, an optimization problem was transferred into a mathematical problem: to find either a local minimum or a local maximum value of *x*. However, this method is not suitable to solve network routing optimization problems, because a network (either for transportation or electrical current) may have hundreds or thousands of variables which results to an impossible number of equations to solve. However, a method has it has been proposed [2.4] to use evolutionary programming to solve network routing problems.

Evolutionary programming (EP) uses biology inspiration inherited from Darwin's theory, which believes that species have the ability to evolve to fit better in their environment in a gradual and natural way [2.5]. A critical presumption of EP is that there exist multiple approaches to a given task, and yet these solutions all stand. The idea is to repeat a certain procedure on a problem until a better or best solution termination condition is met. This is necessary, especially on those puzzles which cannot be solved by conventional methods. EP will choose one of these approaches and match it with the real-world environment.

EP is often referred to as a *population-based optimization*, because it generally consists of a population of many candidate solutions on a specific problem, and as time passes, this population evolves to a better solution. Although very similar to *expert systems* and being able to be classified as a member of the set of artificial intelligence methods [2.6], EP is different to classical expert systems, or fuzzy logic sets, because these model deductive reasoning, while evolutionary algorithms model inductive reasoning.

There are many variants of EP on specific terms. For example, a swarm intelligence algorithm [2.2] is one search function mimicking birds' behavior (bio-inspired intelligence). Ant colony optimization (ACO, [2.3]) is another prominent algorithm, applied largely on Internet search engine design as well. However, some authors declare that they are not part of EP because they only follow specific roles inherited from insect families.

2.2 MULTIPLE OBJECTIVE OPTIMIZATIONS

Recent advancements of EP include the following:

- Simulated Annealing [2.1]
- Ant Colony Optimization [2.3]
- Particle Swarm Optimization [2.2]
- Differential Evolution [2.1]
- Estimation of Distribution Algorithms [2.1]
- Biogeography-Based Optimization [2.6]
- Multi-Objective Optimization [2.7]

Multi-objective optimization is a hot topic in current scientific research. Many real-world optimization problems are multi-objective, especially in engineering design. For example, when wireless network architecture is designed, the engineer needs to consider how many access points need to be placed to cover the whole area. The quantity of these access points is one parameter to optimize, but the coverage of wireless signals is the other factor. Sometimes, the total energy saving of the network needs to be considered as well. The problem of balancing signal coverage and material cost is a multi-objective optimization problem.

2.2.1 Search space and Pareto set [2.7]

A multiple objective optimization problem can be expressed as a function of *x*, where *x* is n-dimensional:

$$\min_x f(x) = \min_x [f_1(x), f_2(x), \dots, f_k(x)] \tag{2.1}$$

Multi-objective optimization process can be treated as a search of a *Pareto optimal set*, which consists of a set of *Pareto optimal points*:

$$x^* \ is \ Pareto \ optimal \ \underset{x}{\Leftrightarrow} \nexists x: \{f_i(x) \le f_i(x^*) \ for \ all \ i \in [1,k], and \ \{f_j(x) \le f_j(x^*) \ for \ some \ \ j \in [1,k]\} \tag{2.2}$$

The goal of a single-objective optimization algorithm is to find an optimal point which is a local minimum (or maximum) value of a certain fitness function. However, in multiple objective optimizations, the goals of an algorithm might be the following:

- Maximize the number of individuals that can be found within a predefined Pareto set;
- Minimize the average distance between a candidate population set and the objective Pareto set;

Sometimes the average distance, M, may be used to evaluate the fitness level of a population. In section 2.3, a multiple objective optimization case is given.

2.3 EP IN DWDM NETWORK DESIGN

In this project, EP was applied as an optimization mechanism after an initial path was given in a geographic map. The idea was to consider a known application of EP to optimize communication channels from previous work in the area within the Warwick Intelligent Systems Engineering Laboratory [2.4]. The further research presented here explored the following:

- What is the performance of an EP-based optimization strategy when the network receives multiple connection requests?
- How does the network behavior change when the Cost 239 configuration is replaced with an ad-hoc scenario (in simulation, a 40 node random network)?

The following sections first introduce this optimization process, and then apply it to two different scenarios. After this, the performance of the strategy is analyzed to ascertain its suitability for more dynamic and random network environments.

2.3.1 Network Model

The benchmark network used in this paper is the Cost 239 Pan European Network (PAN) [2.4]: an optical fiber network connecting major European cities. A diagram of which is shown in Fig 2.1 (Page 25). This network has 18 nodes and 69 links, with an average node degree of 3.83. The weight assigned to each link is the actual physical distance between two cities in kilometers. It is assumed that before any algorithm initiates, no connections exist in the network.

The creation of the dynamic connection is based on the scheduled traffic model previously utilized by Kuri et al. [2.5]. In this approach, the scheduled traffic demands, CR_i, are represented thus:

$$CR_i = [s, d, c, T_s, T_e],\ \ s, d \in V \tag{2.3}$$

where s and d are the source and destination node pair, c is the capacity requested by this traffic (in terms of wavelengths), and T_s and T_e are the establishment and tear-down times of the traffic connection.

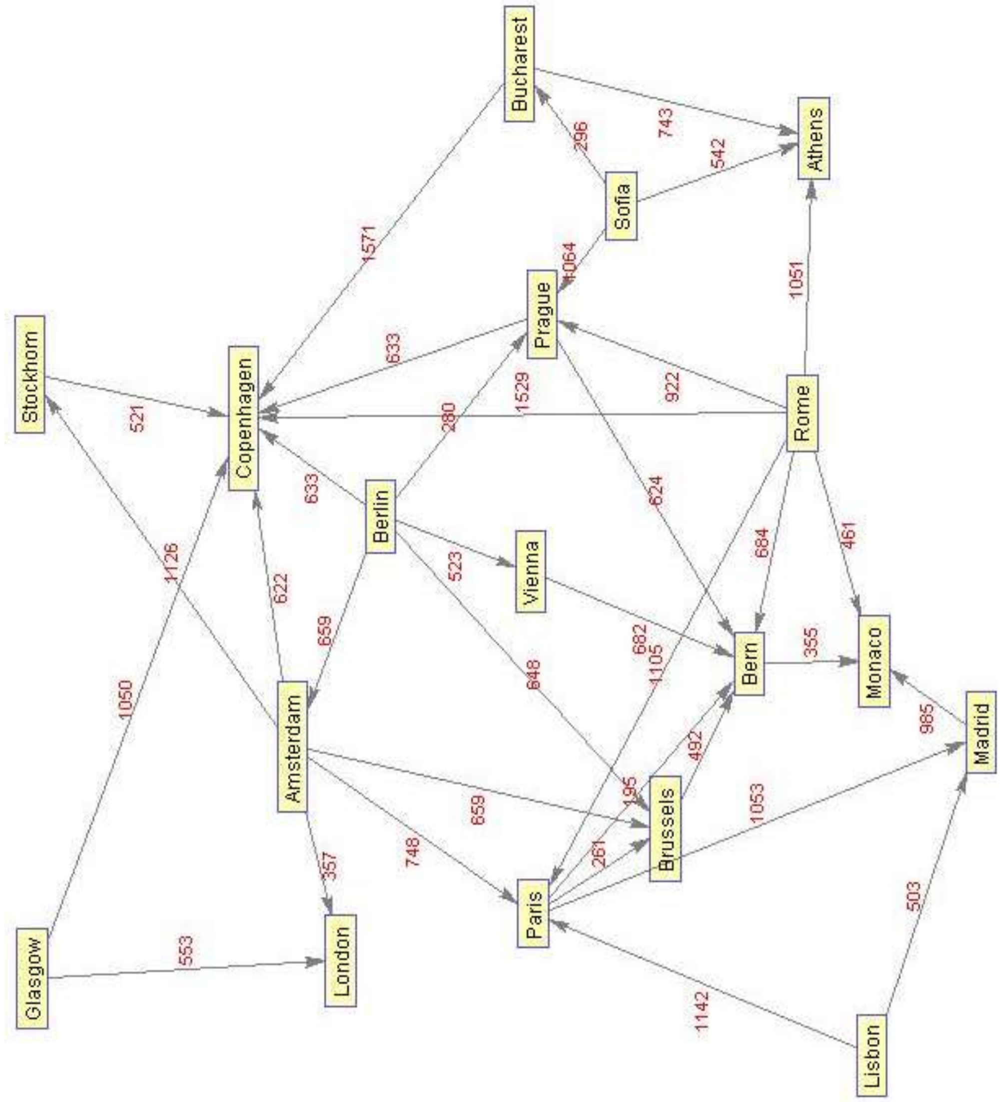


Fig. 2.1 European Cost239 Network—Topology with actual distances in km

Table 2.1 shows an example of a set of traffic requests expressed within this scheduled traffic model.

Figure 2.2 illustrates three connection requests showing both their starting times and their tear down times. In addition, there is a capacity constraint on c that arises both because there are clearly limited numbers of nodes and wavelengths and because it is assumed here that the traffic is unidirectional. The latter reason results in half of the capacity of node s being reserved for traffic reception or equivalently that a restriction on capacity exists such that the allocation for traffic CR_i on node s cannot exceed half of the maximum capacities of all links originating from node *s*. Mathematically, for a number of links L_s connected to node s, each link supports up to K wavelengths [2.5],

$$c \leq K \times L_s/2 \tag{2.4}$$

The links in the COST 239 network are specified in Table 2.2, including, *inter alia*, values for L_s.

In order to avoid blocking, the input process is assumed as follows:

For each node, before its capacity is full, the connection requests that originated from it are treated with "immediate service"—no traffic congestion will be caused from the starting point, but it is possible to get congestion at other nodes; A timer is set for the first connection request so that, once the node becomes congested, this connection request will be dropped—the network links allocated to it will be freed.

Table 2.1 Example of scheduled traffic requests

Connection Requests	s	D	c	Ts	Te
CR0	15	5	6	1	3
CR1	17	3	1	2	4
CR2	3	8	3	3	6
CR3	17	6	4	4	9

A global resource lookup table is used to keep a record of every node's available links. This table is updated as soon as a new connection is assigned links and wavelengths. Once the resource of a certain node is depleted, it will be marked in the table and deactivated in the network.

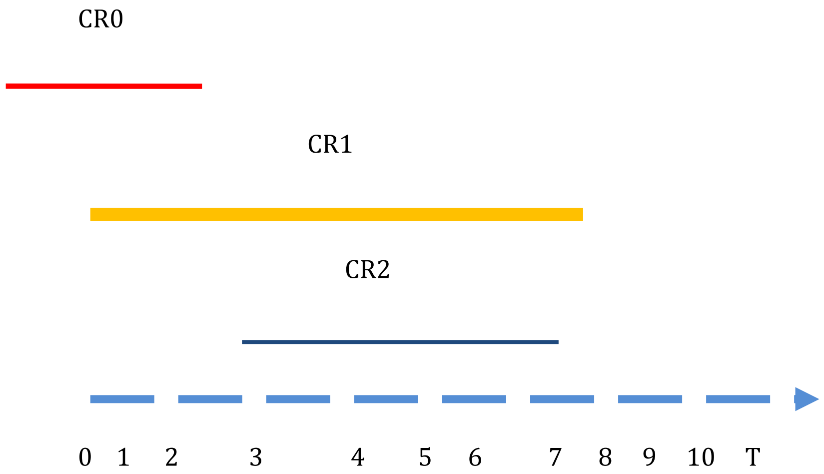


Fig. 2.2 Example of dynamic connection requests

(x-axis stands for time; y-axis stands for different requests)

An optimization process is executed after a certain period (e.g. 10 connections) to make global network resource allocation more efficient. This will also check every node's load capacity and make modifications to balance the load.

Table 2.2: Available links of each node in Cost 239 network

Node ID	1	2	3	4	5	6	7	8
Link No.	5	3	6	6	2	2	5	2
Capacity	20	12	24	20	8	8	20	8
Node ID	9	10	11	12	13	14	15	16
Link No.	5	3	2	2	3	4	7	3
Capacity	20	12	8	8	12	16	28	12

2.3.2 Path routing algorithm

In this section, a dedicated path protection scheme is applied to every existing traffic connection to make sure they are protected in advance. A Dijkstra-shortest-path algorithm [2.6] is used to find a primary route and three backup routes for each community. The backup path is designed with a backup path strategy—the backup path and its associated primary path are fully disjoint.

Thus, full recovery is promised on primary paths for every existing connection in this network.

Table 2.3: Example of Path Set Series

(s,d)pair	1, 17	2, 11	3, 10	5, 4	6, 10	8, 16
Path 1	1-9-17	2-9-15-11	3-15-10	5-16-18-4	6-15-10	8-17-7-16
Path 2	1-14-7-17	2-10-15-3-12-11	3-1-9-2-10	5-7-17-4	6-3-1-9-2-10	8-1-9-17-18-16
Path 3	1-8-17	2-13-4-15-11	3-7-17-4-13-10	5-9-4	6-13-10	8-5-16
Path 4	1-3-14-17	2-12-11	3-6-15-10	5-13-4	6-2-10	8-18-16

Table 2.3 gives examples of path sets and, as may be seen in the table, the search results from the Dijkstra algorithm successfully meet the survivability requirement by providing three backup paths for each connection. However, some backup paths are advantageous in capacity reduction, as well as link load balance. Thus, it is possible to reduce total network capacity usage and balance link loads through the exchange of a primary path and its backup paths.

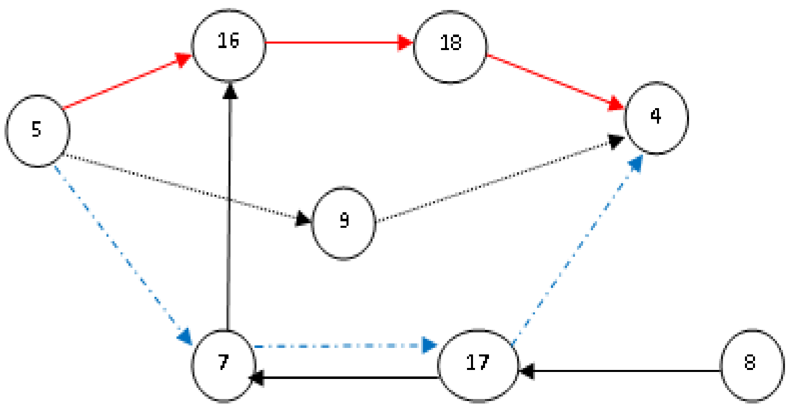


Fig. 2.3 Example of alternative path selection

Fig. 2.3 presents an example of multiple paths and the advantages of alternative path selection. In the figure, red lines denote a primary path located by the

Dijkstra algorithm, whilst the dotted line paths represent backup paths. By inspection, it is easily recognized that the path 5-9-4 has fewer hops compared with the primary path 5-16-18-4, and clearly fewer hops means reduced capacity occupation. Backup path 5-9-4 is a better choice, for traffic where capacity is the primary driver rather than pure transmission speed. Backup path 5-7-17-4 has the same hop length as the primary path but does share the link 7-17 with other traffic increasing the probability of blocking. In summary, backup path 5-9-4 will be chosen for total network capacity optimization, and backup path 5-7-17-4 will not be chosen because of the increased risk of traffic blocking.

2.3.3 Genetic algorithm (GA)

A genetic algorithm (GA) [2.7] is a searching technique to find solutions of optimization problems. It is a class of evolutionary algorithm which uses techniques inspired by evolutionary biology such as inheritance, mutation and crossover. A general process flow chart of a GA is shown in Fig. 2.4, where the GA simulates a problem solving process as a biological evolutionary path by involving population creation, population reproduction, and an offspring selection process. For the path optimization issue, the path set found by the path searching algorithm will be used as an initial population. The GA will cross over different path sets that originate from different connection requests to optimize the global performance of the network.

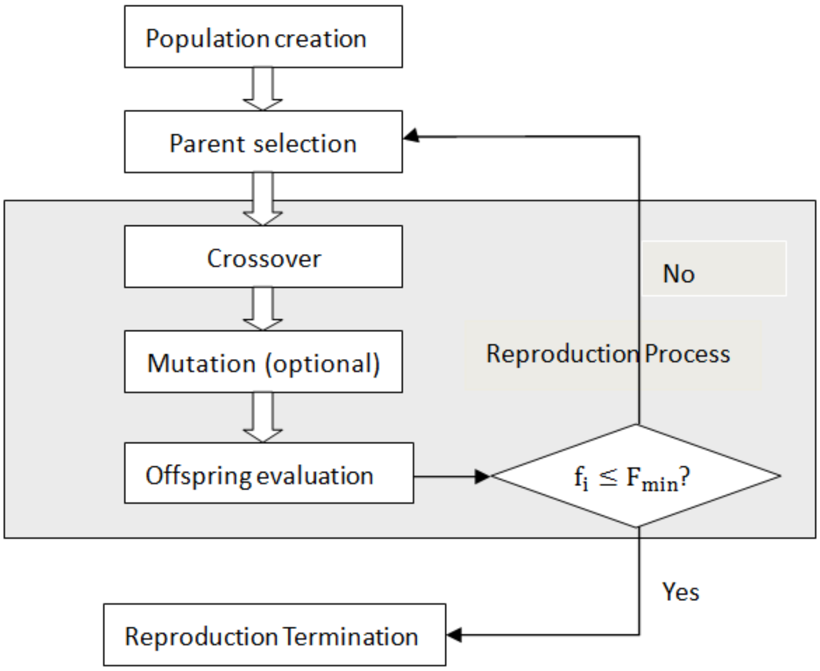


Fig. 2.4 Flow chart of genetic algorithm process

To employ a GA, it is necessary to map the problem to a set of chromosomes that represent the candidate solutions. In this application, the chromosomes (or population) are encoded, as shown in Table 2.4:

Table 2.4 Example of chromosome encoding

0	1	1	0	1	1	0	0	1	1	0	1

CR 1,   CR 2   ....   CR N

Path 2   Path 3   ....   Path n

The chromosome consists of 2*n* digits, where *n* is the number of live connections in a specific timeslot. In other words, 2 digits represent an existing connection in network. The coding scheme represents different path orders, using 00 as the primary path and 01, 10 and 11 as backup paths 1, 2 and 3.

2.3.4 Fitness function

To determine which solutions are the best, it is necessary to define a fitness function dependent on the particular problem. Thus, in the language of optimization, the fitness function is an objective function which calculates a specific outcome (fitness value) from chromosomes. The value that is produced by the fitness function is used as the basis of the decision on whether or not a given chromosome will survive into or produce offspring for the next evolutionary generation. Later, this choice of progression to the next generation will be discussed, but first the optimization strategies employed in this work will be described, along with their specific fitness functions.

A. Least-Loaded Routing (LLR)

This policy defines the fitness function as the total network capacity usage under the traffic pattern contained in the chromosomes. The fitness value of a certain chromosome represents the network wavelength resources allocated to the traffic connections, and the objective is to find a chromosome with the least network usage. The target of this optimization approach is to minimize total network resource capability allocated within all path candidates. The fitness function for capacity c_i and length $l_{i,j}$ is:

$$fitness_m = \sum_i^K \sum_j^m c_i l_{i,j} \tag{2.5}$$

K is the number of existing connections in the network, and m is the number of links contained in the i^{th} chromosome. This function sums the total capacities used on individual links contained in a specified route, and assigns this value to the route (chromosome) as an optimization target. To encompass all routes, the mean of all fitness values from a whole population is used and here this is referred to as the AWUT (Average Wavelength Usage per Traffic), and is defined below. The AWUT forms the target used to evaluate the ultimate optimal results from traffic patterns in the simulations presented later.

B. Least-Congested Routing (LCR)

This policy defines the fitness function as the network blocking probability under the traffic pattern contained in the chromosomes. Here, the fitness value of a certain chromosome represents the probability that the traffic will experience congestion on the current routes, and the objective is to find a chromosome with the least congested route. In this optimization approach, the probability is minimized within all path candidates producing a fitness function:

$$fitness_m = \sum_i^K \sum_j^m p_{i,j} \tag{2.6}$$

$p_{i,j}$ is the probability of congestion of the i^{th} traffic flow congestion on the j^{th} link, calculated as:

$$\mathrm{p_{i,j}} = \frac{\mathrm{c_{i,j}}}{\mathrm{c_{max}}} \tag{2.7}$$

where $c_{i,j}$ is the capacity allocated to the i^{th} traffic flow on the j^{th} link, and c_{max} is the predefined maximum available capacity for the network. This means that as one would expect, the more capacity that is used on a link, the higher the probability will be that it will cause traffic congestion. This fitness function sums the traffic congestion probabilities on individual links contained in a specified route, and assigns this value to the route (chromosome) as an optimization goal. In this case, a target value termed the ANBP (Average Network Blocking Probability) is applied in the simulations below to evaluate the ultimate optimal results from this pattern. The ANBP is the mean value of all LCR fitness values from a whole population, and again its observation permits the monitoring of the evolutionary process.

2.3.5 Crossover and mutation [2.7]:

Crossover is a genetic operator used to vary chromosomes from this generation to next. The operation exchanges the path order of two path sets and generates new path sets for the same traffic. The appropriate fitness function is used to evaluate whether the new path sets will be selected for the next generation.

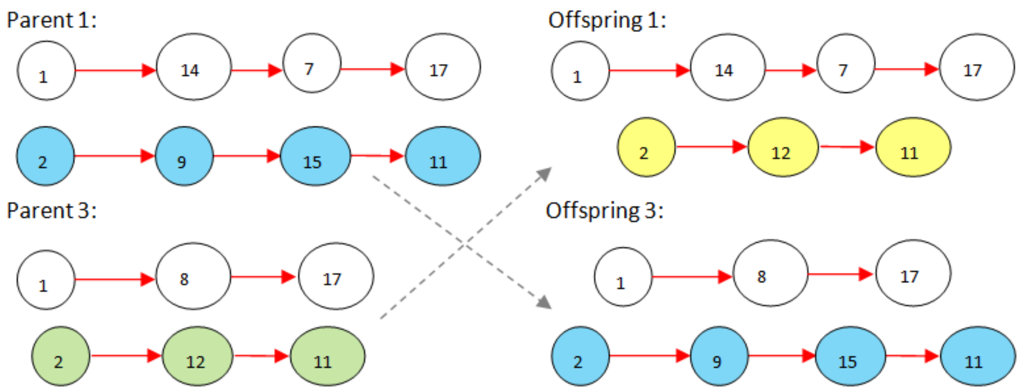


Fig. 2.5: Crossover operation

An example of crossover operation is shown in Fig. 2.5, where paths are allocated to two connection requests (1-17 and 2-11) using the path routing algorithm. After encoding, chromosome parent 1 has the route set 1-14-7-17 for traffic 1-17 and route set 2-9-15-11 for traffic 2-11, while chromosome parent 3 has the route set 1-8-17 for traffic 1-17 and route set 2-12-11 for traffic 2-11. The crossover operation exchanges their second route set and generates offspring 1 and 3. Offspring 1 is fitter than parent 1, since the total wavelength usage is reduced by 1 hop, so it will be selected for the next generation in LLR policy. Also it decreases the congestion of link 2-9, 9-15 and link 15-11, but with an addition of link 2-12 and 12-11, so it is necessary to calculate the global network congestion probability to see if it is better than parent 1 in the LCR policy.

Mutation is the other genetic operator used to maintain genetic diversity from one generation to the next by creating mutated chromosomes. In the simulation algorithm, it is designed as a triggered operation when an evolutionary process enters a dead loop with a group of similar chromosomes. The mutation operation selects the first constant bit, i.e. the position in which all chromosomes have the same value and reverses it.

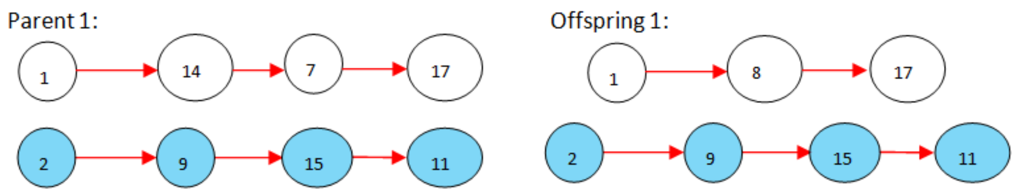


Fig. 2.6 Mutation operation

Figure 2.6 illustrates the generation of a mutated offspring that arises and delivers a shorter route from node 1 to node 8. This operation is beneficial to the solution pool after a series of crossover operations, since it creates a completely novel chromosome that contributes to the diversity of the population.

2.4 SIMULATION RESULTS

Simulation setup

In order to fully address the algorithm's efficiency, the simulation results are presented in two parts. In part one, the performance of the algorithm under a benchmark network scenario is analyzed. In part two, the algorithm is compared with two other conventional routing assignment techniques under different network environments and with different traffic loads to illustrate better a genetic algorithm's advantages and disadvantages. All simulations were run on Windows XP with a 3.31 GHz CPU and the software platform was MATLAB (version: 7.8.0.347, R2009a).

Part one: single network performance

Ten iterations of randomly-generated traffic requests were chosen as an initial data set for the routing and optimization algorithms.

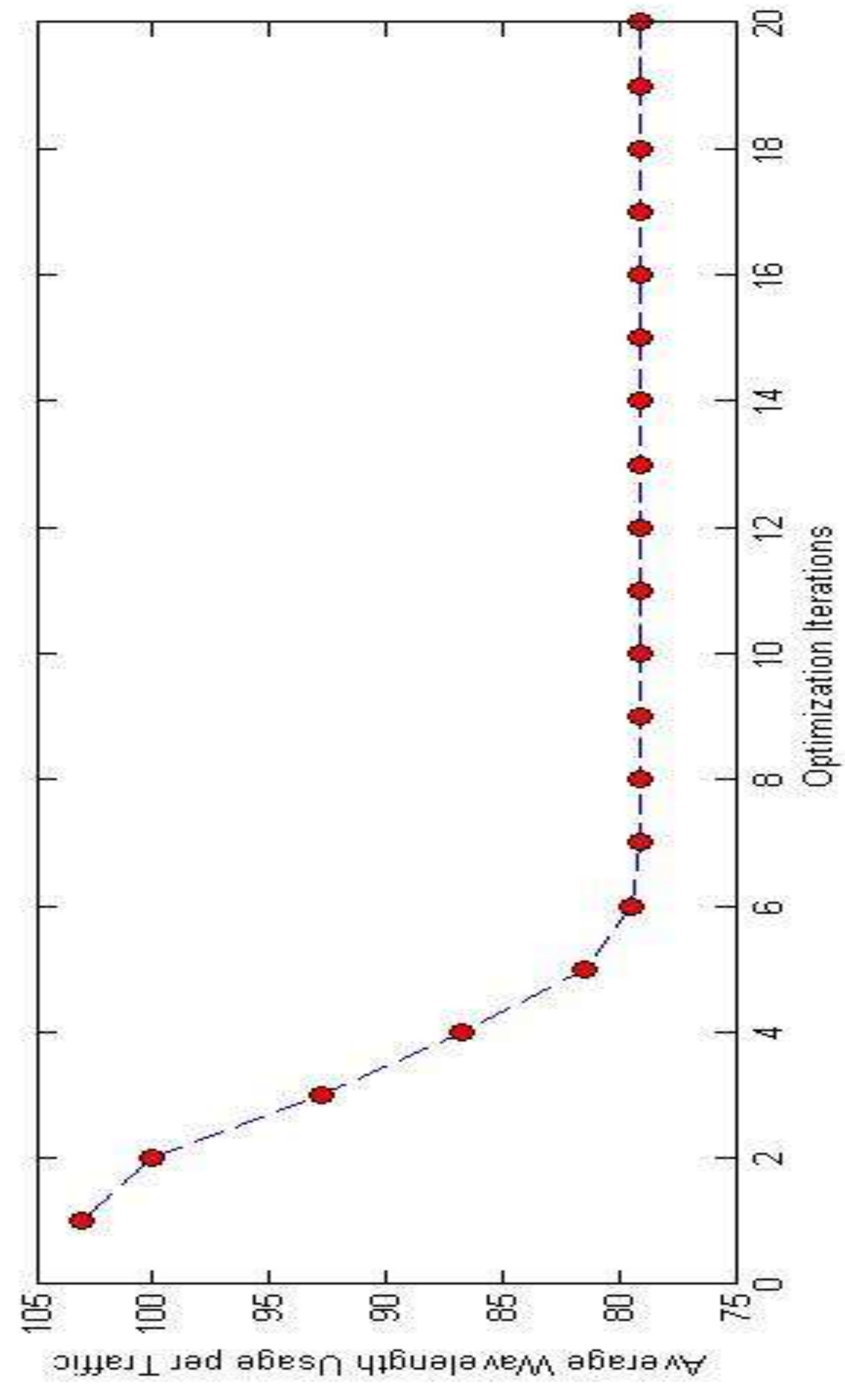


Fig. 2.7 AWUT optimization result

Each of the iterations contained ten connection requests with exactly the same format as that shown in Table 2.1. The optimization results and observations concerning them are presented below, along with the ALR (Actual output versus Limit Ratio) data record.

Average Wavelength Usage per Traffic (AWUT) optimization

As discussed before, the objective of the optimization algorithm within the LLR policy is to minimize the global network resource usage, and AWUT is a performance criterion to reflect the algorithm efficiency. AWUT is defined as the following:

$$\mathrm{AWUT} = \frac{\sum_{i}^{N} C_i \times L_i}{N} \tag{2.8}$$

where N is the number of traffic streams existing in the network, C_i is the capacity of i^{th} stream, and L_i is the link number of the primary route assigned to the i^{th} traffic stream. The AWUT represents the average network resources (lightpaths/wavelengths) allocated per traffic stream in the primary path assignment. From Fig.2.7 (page 36) it is clearly shown that, after 6 iterations, the AWUT has decreased by 23.3% (from 103 to 79) and then has become stable. In this case, the lower limit is 79, and the algorithm successfully reaches its optimal outcome. An advantage of this optimization process is that it only needs traffic information as input, and calculates the optimal paths according to local information. Compared with adaptive routing algorithms which need support from the network signaling layer and an appropriate protocol in advance, the GA simulates an environment in which best paths naturally evolve from a group of candidates, which can be calculated and stored by routers in advance.

ALR (Actual outcome versus Limit Ratio) optimization

ALR is a performance criterion for the optimization achievement, and it is defined as the following:

$$\mathrm{ALR_i} = \frac{\mathrm{Initial_i} - \mathrm{Actual_i}}{\mathrm{Initial_i} - \mathrm{Min_i}} \tag{2.9}$$

where ALR_i is the ALR value for i^{th} iteration of optimization. $Initial_i$ is the i^{th} initial total network load, $Actual_i$ is the i^{th} optimization result achieved by the algorithm, and Min_i is the i^{th} lower limit of the algorithm known in advance.

Apart from several optimal cases, ALR (as shown in Fig. 2.8) is a constant value ranging from 70% to 90% for most of the optimization rounds. To achieve this value the algorithm usually takes 8 to 12 iterations to optimize the path. Compared with the simulation results in [19], the static optimization case, the dynamic algorithm's single-iteration performance (Fig. 2.7) is superior (23.3% versus 7.1%), but the average performance over a long time is similar. The optimization performance of the GA relies heavily on the nature of the path set—its population, and the higher the deviation the path sets have, the better the optimization result the GA can achieve.

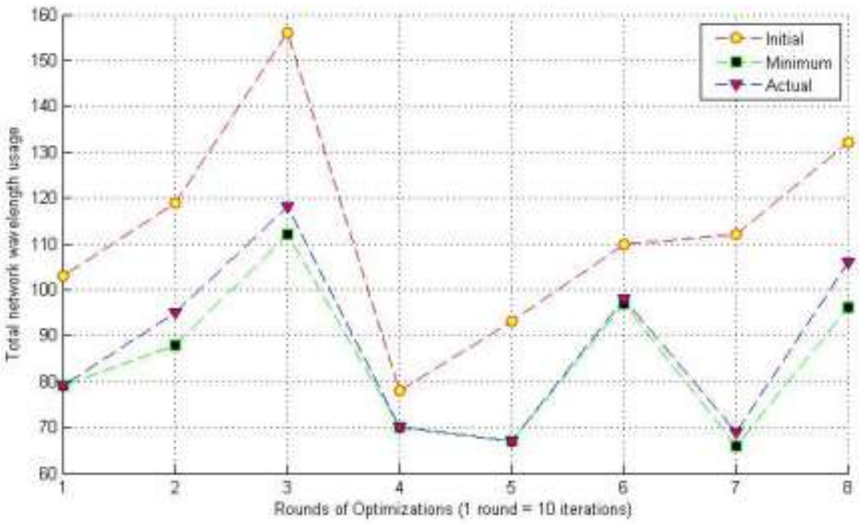


Fig. 2.8 ALR result for 8 iterations

However, the advantage of the GA is apparent in that the number of iterations needed to reach the optimal value, given a certain network topology, is almost constant, no matter what the nature of the traffic is, and what complexity its route involves for calculation. The optimization speed depends on the crossover population size versus the total population size. In the simulation environment, the crossover population size is 20, and the total population size is $4K$, where K is a variable, represents existing connection numbers. In case of Fig. 2.7, where $K = 7$, the population size is 16384, then the GA optimization has the same effect with 2730 ordinary ranking and sorting operations, and a single crossover operation has the same effect with 273 such operations. From this point of view, implementing GA will greatly accelerate the speed of path routing in DWDM network.

Traffic Blocking Probability (TBP) optimization

TBP is another optimization objective within the LCR policy, and its mathematical expression is as follows:

$$TBP = \max(BP_i)\,, i \in [1, K] \tag{2.10}$$

where BP_i is i^{th} route blocking probability:

$$BP_i = \max \frac{c_l}{c_{max}}, l \in [1, L_i] \tag{2.11}$$

where c_l is allocated capacity (in wavelength) of the l^{th} link in i^{th} route, and c_{max} is the maximum capacity of each link. The traffic blocking probability is the maximum value among its individual traffic blocking probabilities which are presented by peak values from their link blocking probabilities.

The calculation of the TBP makes use of a global resource lookup table (mentioned in dynamic connection creation, section II: Methodology), which keep records of each link's wavelength allocation status. The link is expressed as:

$$L_i = \{(s, d), C_i\} \tag{2.12}$$

where s and d represent the source and destination pairs for link L_i, and C_i is the allocated link capacity. Fig. 2.9 gives an example of a TBP optimization result:

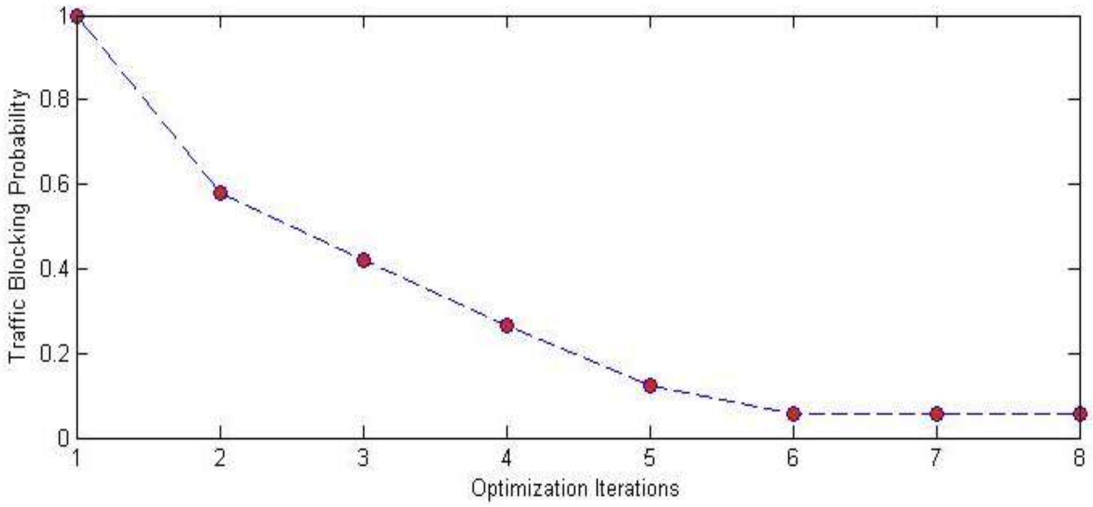


Fig. 2.9 TBP optimization result after 8 iterations

It is worth noting that the value of the TBP not only serves as a warning for individual links when they reach maximum capacity, but also represents a capacity occupation level for the entire network, given the in situ capacity allocation condition for all links. If the value of the TBP is below 0.5, the network should to be in good condition, with more than half link capacities being free for future traffic. An effective operation in TBP reduction (as shown in Fig. 2.9) is to reallocate traffic from more congested paths to less occupied paths, thus achieving an overall optimal result. From Fig. 2.9 it may be seen that the simulation successfully decreased overall TBP from 1 to 0.1 after 7 iterations.

Part two: comparative results

In order to fully evaluate the performance of the GA under different network scenarios, a 40 node Gaussian distributed network was created (shown at Fig. 2.10) as a counterpart to the Cost 239 network. Two commonly-used searching techniques were used as benchmarks:

- First-fit (FF) [2.8]

- Directly Relative Capacity Loss (DCRL)[2.9]

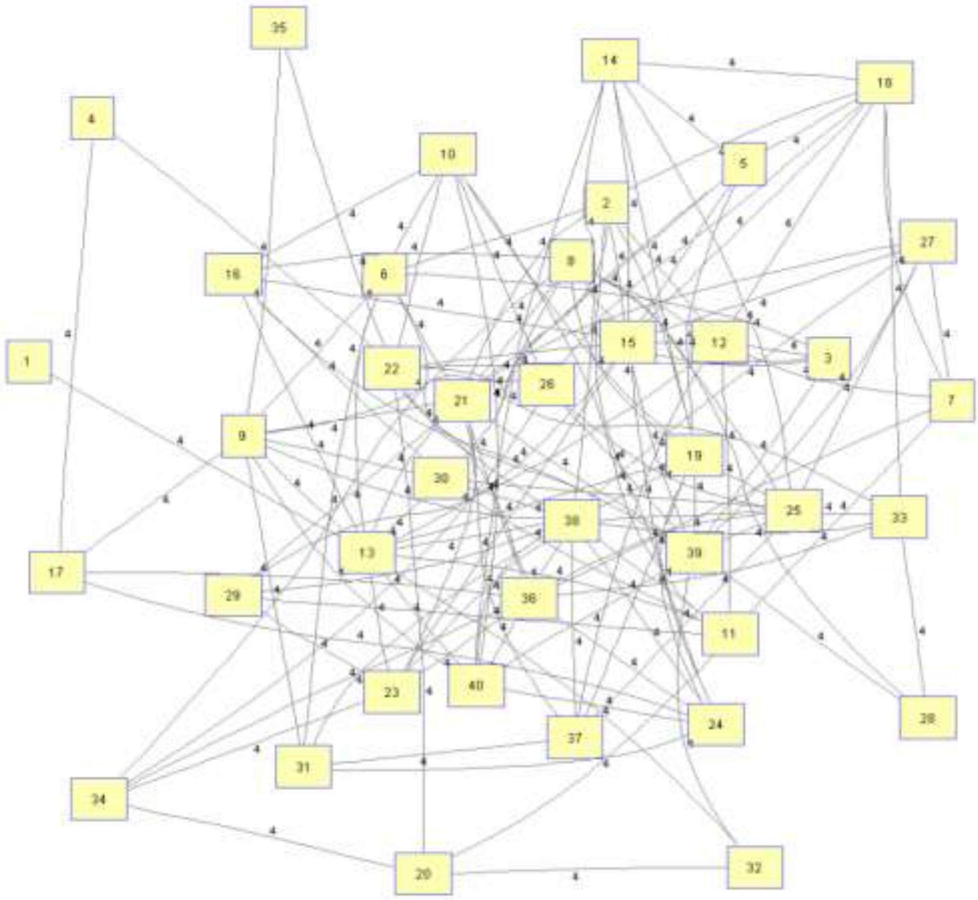


Fig. 2.10 Gaussian distributed 40-node network

The FF strategy simplifies the wavelength assignment procedure into a “first come first serve” process: the first incoming traffic is assigned the first available wavelength within the wavelength index, and when no more wavelengths are available, the traffic is blocked. In the DRCL strategy, each node stores information on the capacity loss on each wavelength in its own lookup table. This table is called the Relative Capacity Loss table and whose content is a triple of wavelength, destination, and RCL weight. When a connection request arrives and more than one wavelength is available on the light path, DRCL will make use of the RCL table and calculate the path which has a minimum RCL weight among its peers.

Table 2.5 comparison of FF, DRCL and GA using four wavelengths and uniformly distributed traffic

	Blocked Nodes			Blocked Paths			Consumed Wavelengths			Elapsed Time		
	FF	DRCL	GA	FF	DRCL	GA	FF	DRCL	GA	FF	DRCL	GA
Cost 239	1/18	0/18	0/18	27/69	20/69	26/69	120/276	109/276	123/276	0.2304 s	0.0168 s	70.7627 s
Network	2/18	2/18	0/18	31/69	28/69	27/69	132/276	124/276	126/276	0.1795 s	0.0670 s	59.4942 s
	2/18	2/18	0/18	31/69	29/69	27/69	133/276	126/276	126/276	0.1931 s	0.0652 s	62.5776 s
	FF	DRCL	GA	FF	DRCL	GA	FF	DRCL	GA	FF	DRCL	GA
Random	6/40	12/40	6/40	53/136	22/136	43/136	333/544	360/544	293/544	1.4306 s	0.2976 s	105.1816 s
40-node	18/40	17/40	22/40	109/136	89/136	107/136	504/544	488/544	487/544	1.6792 s	0.6016 s	103.3572 s
Network	26/40	25/40	23/40	112/136	109/136	108/136	529/544	512/544	501/544	1.7709 s	0.6110 s	96.2361 s

The wavelength constraint is set as 4 for each edge: this means each edge can be allocated with a maximum of 4 wavelength channels. The network traffic is set with 3 groups to simulate different network conditions (no congestion, light congestion and heavy congestion), and they are set as uniform traffic at the beginning.

The simulations are set to compare four parameters: blocked nodes, blocked paths, consumed wavelengths and elapsed time. The first three parameters are used to evaluate algorithm's efficiency to allocate traffic requests on network and the last parameter is used to evaluate the algorithm's execution efficiency.

Table 2.5 (page 43) indicates that when network bandwidths (wavelength numbers) are constrained and input traffic is uniformly distributed, the GA is not the ideal choice for network optimization. The capacity limitation severely affects the natural evolution process of the GA algorithm in that limited network resources are exhausted very quickly by incoming traffic. The uniformly distributed traffic pattern also affects GA efficiency because individual traffic streams do not differ much from each other, thus making natural evolution difficult for producing a better generation. On the other hand, the GA spends 100 times the CPU time for optimization that FF does. This case is a striking example of the advantage of simple strategies, such as FF and DRCL in this simple and somewhat unrealistic network scenario.

However, when DWDM applies on the network, there are many more channels available, for example the OC-64 standard supports 64 wavelengths and the OC-128 standard 128 wavelengths in one channel. This dense allocation of wavelengths begins to give the GA an advantage over a simpler scheme. Table 2.6 below shows the results with uniform traffic and a network whose channel supports 64 wavelengths.

Table 2.6 comparisons between FF and GA using 64 wavelengths and uniform traffic (DRCL results are similar to FF so not included)

	Blocked Nodes		Blocked Paths		Consumed Wavelengths		Elapsed Time	
	FF	GA	FF	GA	FF	GA	FF	GA
Cost 239 Network	0/18	0/18	0/69	0/69	291/4416	268/4416	0.1692 s	58.6345 s
	0/18	0/18	0/69	0/69	574/4416	566/4416	0.1616 s	62.3060 s
	0/18	0/18	2/69	0/69	861/4416	868/4416	0.1559 s	58.0856 s
Random 40-node Network	FF	GA	FF	GA	FF	GA	FF	GA
	0/40	0/40	0/140	0/140	403/8960	372/8960	1.1071 s	105.1816 s
	0/40	0/40	0/140	0/140	823/8960	768/8960	1.0644 s	103.3572 s
	0/40	0/40	0/140	0/140	1226/8960	1164/8960	1.0678 s	96.2361 s

Table 2.7 Comparison between FF and GA using 64 wavelengths and randomly distributed traffic

	Blocked Nodes		Blocked Paths		Consumed Wavelengths		Elapsed Time	
	FF	GA	FF	GA	FF	GA	FF	GA
Cost 239	0/18	0/18	1/69	0/69	1409/4416	284/4416	0.2083 s	57.4625 s
Network	0/18	0/18	5/69	0/69	1939/4416	471/4416	0.1868 s	57.4673 s
	0/18	0/18	9/69	0/69	2079/4416	532/4416	0.1900 s	58.3830 s
	FF	GA	FF	GA	FF	GA	FF	GA
Random	4/40	0/40	1/140	0/140	2363/8960	442/8960	1.0979 s	105.9036 s
40-node	4/40	0/40	6/140	4/140	4333/8960	796/8960	1.2802 s	103.3572 s
Network	14/40	0/40	15/140	11/140	6069/8960	1346/8960	1.5023 s	96.2361 s

Finally, the simulation performance was testing in a dynamic environment, where traffic is randomly distributed to simulate a variable data input situation. The simulation results in Table 2.7 show that in this scenario the GA comes into its own. On the Cost239 network the GA saves between 25-35% of the network capacity compared to FF; for the random 40 node network, the savings are approximately 20-50% over FF. Moreover, the GA also produces fewer blocked paths.

2.5 CONCLUSIONS

In this chapter, an introduction is given to evolutionary programming, which is a technique to be applied on optimization problems. In order to utilize EP on VANETs, a starting point derived from previous work in the Intelligent Systems Engineering Laboratory at Warwick was utilized. This has applied EP to optimize network resource usage on a DWDM network, and had largely focused on the static network case with dynamic network performance unexplored. Here, since VANETs comprise dynamic elements, section 2.3, included a simulation of EP capabilities for a dynamic network case. This demonstrated that this was the most advantageous environment for a GA, which significantly outperformed a First Fit approach. However, this was bought at the cost of increased calculation time, especially when the network scale is larger than 100 nodes. In summary, EP is suitable for high-performance requirement scenarios, and is very efficient on small to medium networks.

REFERENCES

2.1. S., Dan., Evolutionary Optimization Algorithms, Hoboken: Wiley. ISBN: 9781118659502 1118659503. 2013.

2.2. R. Fletcher, Practical Methods of Optimization (2^{nd} edition), Wiley, ISBN: 9780471915478. 2012.

2.3. E. Oki, Linear Programming and Algorithms for Communication Networks: A Practical Guide to Network Design, Control, and Management, CRC Press, ISBN: 978-1-4665-5264-7. 2012.

2.4. Y. S. Kavian , W. Ren, M. Naderi, M. S. Leeson and E. L. Hines, 'Survivable wavelength-routed optical network design using generic algorithms', European Transaction on Telecommunications, 19 (3), 247 –255. 2008.

2.5. P. Angelov, D. P. Filev, N. Kasabov, Evolving intelligent systems: methodology and applications, John Wiley, 2010.

2.6. E. Richard, Contemporary artificial intelligence, Taylor & Francis. ISBN: 9781439844694. 2013.

2.7. A. Ben-Tal, Characterization of Pareto and lexicographic optimal solutions, in Multiple Criteria Decision Making Theory and Application, vol. 177 of Lecture notes in Economics and Mathematical Systems. Berlin, Germany: Springer-Verlag, pp. 1-11. 1980.

2.8. J.Kuri, N.Puech, M.Gagnaire, E.Dotaro, R.Douville, "Routing and wavelength assignments of scheduled lightpath demands", IEEE Journal on Selected Areas in Communications 21(8) 1231-1240. 2003.

2.9. E. W. Dijkstra. "A note on two problems in connexion with graphs". Numerische Mathematik 1: 269–271. 1959.

2.10. J. H. Holland, Adaption in Natural and Artificial Systems. The University of Michigan Press, 1975.

2.11. A. Mokhtnr and M. Aziroglu, "Adaptive Wavelength Routing inAll-Optical Networks,'' IEEE/ACM Transactions on Networking. vol.2, pp.197-206, April 1998.

2.12. A. Birman and A. Kershenbaum, “Routing and Wavelength Assignment Methods in Single-Hop All-Optical Networks with blocking”, INFOCOM 95, vol.2, p 431-438. 1995.

CHAPTER 3

STOCHASTIC GEOMETRY

3.1. INTRODUCTION

This chapter presents a brief and targeted literature review on stochastic geometry, which is another technique applied to the data sets in this research. Stochastic geometry is a statistical approach to analyze point processes on multidimensional Euclidean spaces, and it is widely used in image processing, biomedical scanning and geographic map analysis. One of the most emphasized point processes is the Poisson Point Process, and so the characteristics of this process are introduced in section 3.2. Then, applications of stochastic geometry in wireless communication networks are reviewed in section 3.3. Compared with static modeling techniques, stochastic geometry offers a simple and flexible way to analyze large scale wireless networks.

In many technical fields such as biology, chemistry, and geography, the data sets to be analyzed are in the form of images. This large number of data points needs an effective way to analyze it. A major question to answer is how to reduce the sheer quantity of these data sets and express them in simple, concise mathematical formations. The subject of transforming images into mathematical forms is called mathematical morphology, and is well established in the literature [3.1]-[3.4]. In this process, images are transformed into pixels, and these pixels are reduced into numbers (such as greyscale). Finally, these images are represented by different distributions of pixels, or even more simply, different parameters of a certain distribution model (such as Gaussian) so as to greatly reduce the data storage needs of images.

The same operation can be immediately applied to digital city maps: once city map information is scanned from satellites or collected from digital sensors, it is stored in a form of images. The only difference is the information is geographic data (such as latitudes), not pixels. However the images can be processed in the same way. A common attribute is the geographic distance, or street length. Adapting these distance data sets into a certain statistical model is a process that is named Stochastic Geometry (SG) [3.6]. In mathematical form, the original data sets are called spatial point sets; the attribute to be fitted is called a measure; the statistical model is called a point process.

The application of SG to wireless networks is relatively new [3.12] but using point processes to represent wireless networks really started with the work of Gilbert [3.5], who analyzed the connectivity of large fixed wireless networks by means of SG. Compared with a static city map, a wireless network is more dynamic and unpredictable. Thus, to apply SG on wireless networks needs the assistance of statistic knowledge of the real situation. One solution is to choose point density as a measure, because if the composition of nodes fits a Poisson distribution, then the point density of its area is a constant value. A great deal of research has been based on this approach [3.7] [3.8], and a detailed analysis is presented in section 3.3.

Finally, compared with evolutionary programming (EP)—the first technique applied in this research, SG has its advantages and disadvantages; table 3.1 compares the two techniques. From table 3.1 it may be observed that EP is advantageous for network modeling, and SG is advantageous in data composition and operation cost. In summary, EP is capable of providing precise and point-to-point connection advice, but, when the network scale is large, or customers requires quick responses, SG is a suitable candidate.

Table 3.1 EP versus SG

	Network modeling	**Data composition**	**Mathematical Analysis**	**Operation Cost**
Evolutionary Programming	Precise; Static;	Very complex	Biology inherited;	Very long
Stochastic Geometry	Fuzzy; Static;	Simple; Fuzzy;	Clear; Concise;	Quick

3.2. POISSON POINT PROCESS (PPP) [3.6]

As reviewed in section 3.1, the basic elements of SG are points: in images, they are pixels; in physics and chemistry, they are elementary particles; on geographic maps, they are street junctions. Random point patterns are widely used in scientific research, and as one of them, the Poisson point process (PPP), has been a focus of particular attention because of its superior properties. A history of the PPP can be found in Guttorp and Thorarinsdottir's work [3.7]. However in my research, a homogeneous Poisson point process is used to simulate sensor network architecture in the US city of Boston. A review of the fundamental properties of the Poisson model is in section 3.2.1 followed by exploration of the method to generate a Poisson point distribution. With this knowledge of the Poisson distribution, section 3 examines application of existing wireless network architectures.

3.2.1. BASIC PROPERTIES

A PPP is a type of point process, which can be visually depicted as a series of points scattered in space. In mathematics, a point process is defined as a measurable mapping Φ on a space E, definite as in equation 3.1:

$$\Phi = \sum_i \delta X_i \tag{3.1}$$

A point process is called a Euclidean space process when the space, E, is the Euclidean space, $\mathbb{R}^d$, of d dimensions. The intensity measure, Λ, of Φ is defined as Λ(B) = EΦ(B). A point mapping Φ is a homogeneous Poisson one [3.6] if:

1. For all disjoint subsets B1, B2, …, Bn of E, the random variables Φ(Bi) are independent with each other;
2. For all sets B of E, the random variables Φ(A) are Poisson distributed as (3.2);

   $$P(N(B) = k) = \frac{\Lambda(B)^k}{k!} \exp[-\Lambda(B)], k \in \Phi, B \in E \tag{3.2}$$
3. The density of the points on space E is constant.

The property in 1) is called independent scattering, while another characteristic in 2) is called Poisson distribution of point counts. The first property is also called 'completely random' or 'totally random'. The second property in 2) is called the intensity or density of a Poisson point process.

3.2.2. POISSON POINT SET GENERATION

Poisson point sets can be generated based on this mathematical property:

Conditioning and binomial point process

Suppose Φ is a homogeneous Poisson point process. Consider a subset, W, of Φ under a condition that Φ(W) = n, where n is the number of points contained in W. Now one assertion that can be made in that W is a finite point process, and it is a binomial point process with n points. Based on this criterion, the simulation of a homogeneous Poisson point process in an area transforms into a task of two stages:

1. Determining the number of points contained in W by simulation with a variable which is Poisson randomly distributed;
2. Defining the position of these points by simulating a binomial point process in W.

In order to generate a homogenous PPP in an area from [0,0] to [A, B], with a pre-defined point density λ, the following steps need be to executed:

1. Total number of points contained in this area is n, where $\prod_{i=1}^{n} x_i < e^{-\lambda}$;
2. Place n number of points randomly within a range [0, A] and [0, B], respectively;

In Fig. 3.1, it may be observed how a PPP is used to simulate the real-world wireless network.

3.3. WIRELESS NETWORK ANALYSIS

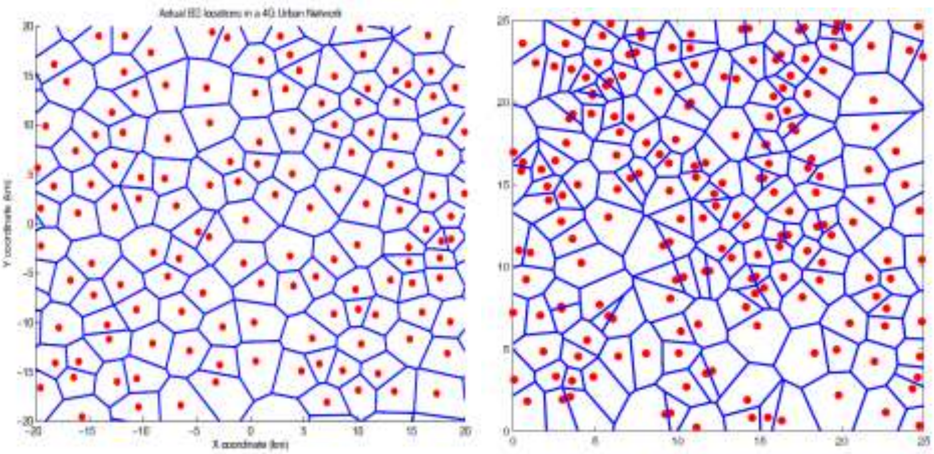


Figure 3.1 (a) A 40×40 km view of a current base station deployment by a major service provider in a relatively flat urban area; (b) a Poisson distributed base stations, with each mobile associated with the nearest base station [3.8]

In order to apply stochastic geometry on wireless networks, an assumption is firstly made that base station distribution follows a homogeneous PPP of intensity λ in the Euclidean plane. In Figure 3.1, both a map consisting of Poisson distributed base stations (left) and the current base station deployment by a major service provider (right) is given. One can observe from the figure that in (b) base stations are closer to each other than in (a). However, by selecting a subarea of (b), for example the central area, base stations will be more evenly distributed. Moreover, graph (b) will allow SG tools to be employed for mobile communications analysis [3.8].

The major objective of wireless network design is to define bounders of base station coverage. In SG, this is called the probability of coverage, and is defined as:

$$\mathrm{p}_c(\mathrm{T}, \lambda, \alpha) \triangleq \mathrm{P}[\mathrm{SINR} > T]; \tag{3-3}$$

Since the data loss rate being explored encompasses multiple hops, this probability of coverage is now a data loss rate probability of a source VANET node in presence of all other peers. When the transmitting channel is free of interference (I = 0) and path loss rate is exponential to the distance (general fading), and the fading coefficient is equal to 4 (α = 4), an expression of signal channel data loss rate formula results [3.8]:

$$p_c(T, \lambda, 4) = \frac{\pi^{\frac{3}{2}\lambda}}{\sqrt{\frac{T}{SNR}}} \exp\left(\frac{\left(\lambda\pi\Re(T)\right)^2}{\frac{4T}{SNR}}\right) Q\left(\frac{\lambda\pi\Re(T)}{\sqrt{\frac{2T}{SNR}}}\right)$$
$$\Re(T) = 1 + \sqrt{T}\left(\frac{\pi}{2} - \arctan\left(\frac{1}{\sqrt{T}}\right)\right) \tag{3.4}$$

The Q function in Equation (3.4) is a standard Gaussian Tail probability function. Once the single hop signal loss rate formula is derived, a Galton-Watson process [3.11] is applied to calculate the multi hop signal loss rate. A complete definition of a Galton-Watson process and its extinction probability is given in by:

$$X_{n+1} = \sum_{j=1}^{X_n} \varepsilon_j{}^n, X_0 = 1, \lim_{n\to\infty} p(X_n) = 0 \qquad (3\text{-}6)$$

If ε_j follows a Poisson distribution, a particularly simple relationship can be found between X_n and X_{n+1} as defined in Equation 3.7:

$$X_{n+1} = e^{\lambda(X_n - 1)} \qquad (3\text{-}7)$$

Now that multi-hop transmission process has been converted into a Galton-Watson process with an attached survival probability X_n, a final expression of relationship between the Probability of Signal Loss Rate and the peer-to-peer transmission error is given in Equation 3.8 [3.8].

$$p_{e,k+1}(\mathrm{T}, \lambda\,\mathrm{T}, = \left(1 - \frac{\mathbf{1}}{\mathbf{1} + \sqrt{\frac{\mathbf{T}}{\mathbf{log}\left(\frac{\mathbf{1}}{\mathbf{1}-\mathbf{p_{e,k}}}\right)}} \times \mathbf{arctan}\left(\sqrt{\frac{\mathbf{T}}{\mathbf{log}\left(\frac{\mathbf{1}}{\mathbf{1}-\mathbf{p_{e,k}}}\right)}}\right)}\right) \times \mathbf{e}^{\boldsymbol{\lambda}\times(\mathbf{p_{e,k}}-\mathbf{1})} \qquad (3\text{-}8)$$

Figure 3.2 shows a curve which relates the number of hops of which a signal link consists and its probability of extinction as a function of the frame error rate (FER) required. It should be noted that this probability is $1\text{-}X_n$ and so it starts from 0 and approaches 1 s n approaches infinity.

$$p_{e,k+1}(\mathrm{T},\lambda,4)=\left(1-\frac{1}{1+\sqrt{\frac{\mathrm{T}}{\log\left(\frac{1}{1-p_{e,k}}\right)}}\times\arctan\left(\sqrt{\frac{\mathrm{T}}{\log\left(\frac{1}{1-p_{e,k}}\right)}}\right)}\right)\times e^{\lambda\times(p_{e,k}-1)} \tag{3-9}$$

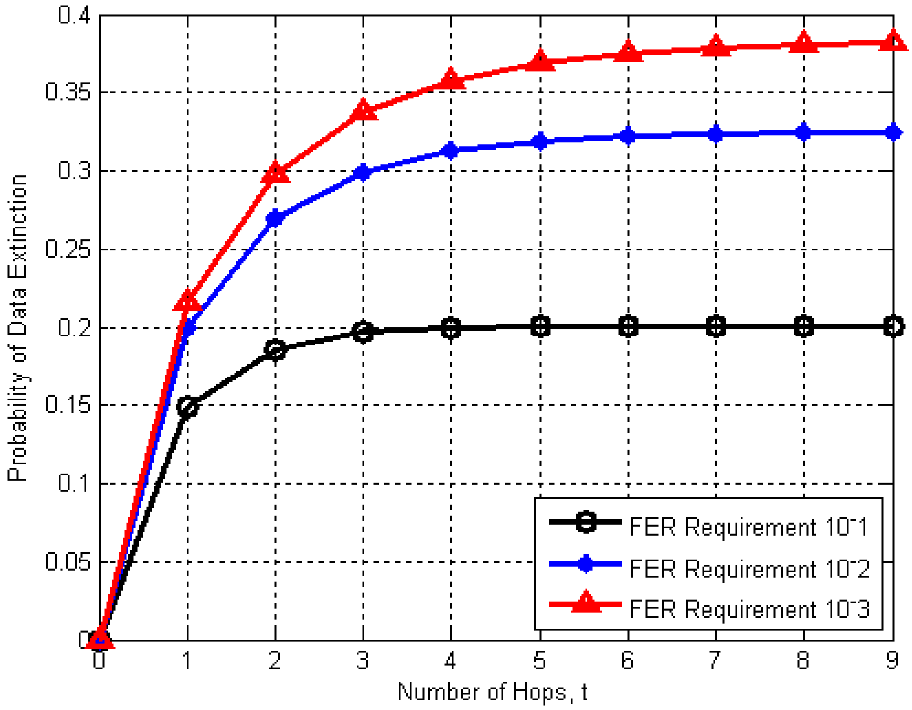


Fig. 3.2 Diagram of multi-hops and probability of data extinction

3.4. BOSTON CITY SCENARIO

In order to integrate SG theory into the Boston city scenario, the following steps need to be executed:

1. Define a measure (node density; street length, etc.) for Boston city;
2. Divide the Boston city map into different areas, where inside the measure could be constant or nearly constant;
3. Calculate the value of measure in each area, then translate it into λ by using formula $\lambda = u_i \times v_i, i = 1,2,\dots,n$; in this formula, i is the area

index ranging from 1 to n; u_i is the value of measure (e.g. node density) in area i calculated in advance; v_i is the volume of area i (e.g. if area i is a square of A meters high and B meters long, then $v_i = A \times B$).

4. Apply λ in equation (3-8) to calculate data loss rates of each area.

For example, adopting node density as a measure for the Boston city map, then Boston city can be divided into seven disjoint areas, as shown in Fig. 3.3:

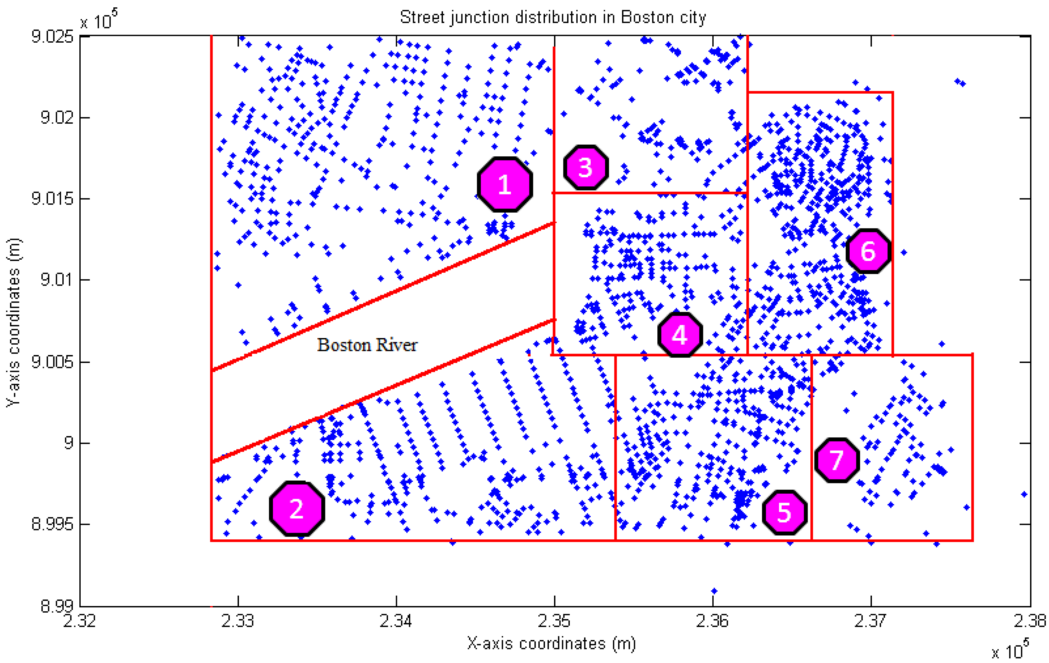


Figure 3.3 Area division of Boston city

In contrast to a MANET, the nodes in Figure 3.3 are street junctions, because in VANETs the base stations (or wireless transmitters) are placed in street junctions to assist signal transmission. The positions of these transmitters can be obtained in advance through a digital map.

After the areas are divided, a probability distribution (PDF) needs to be calculated for each area. The PDF for first 6 areas are plotted in Figure 3.4.

With these PDFs available, it is possible to calculate λ of each area and the results are shown in table 3.2.

Table 3.2 Values of Poisson model λ in seven subareas

Subarea	1	2	3	4	5	6	7
λ	0.5278	0.4776	0.3793	0.5584	0.9412	0.8	0.32

When λ values of all areas are obtained, a virtual Poisson graph is constructed for further analysis; a sample Poisson graph is shown in Figure 3.5.

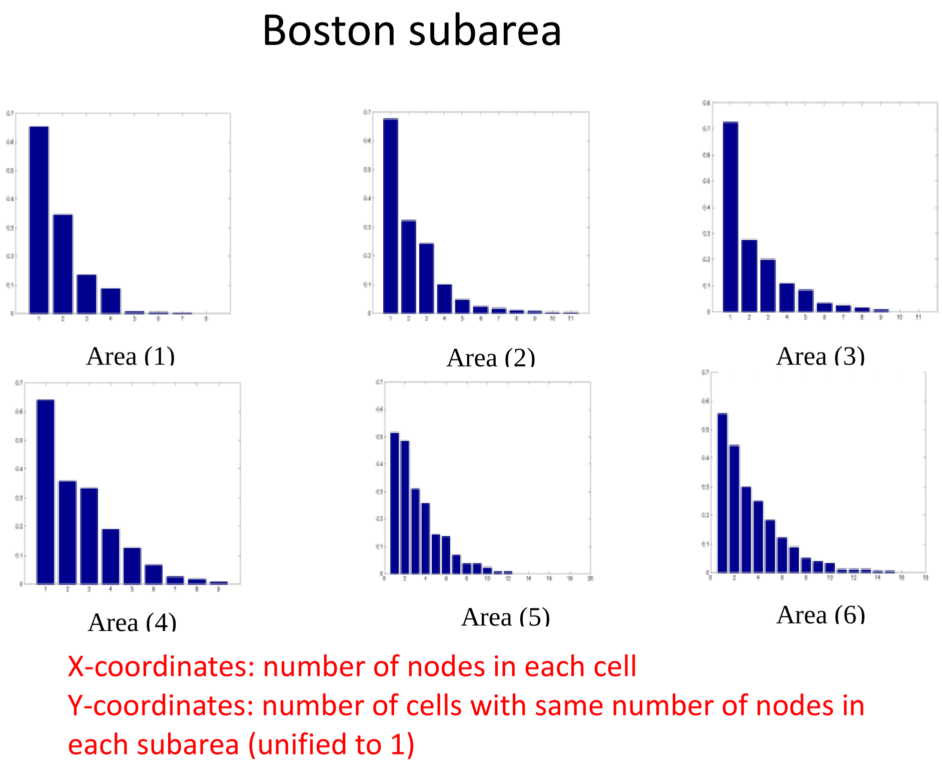


Fig. 3.4 Probability density distributions (PDF) of 6 areas labeled in Fig. 3.3

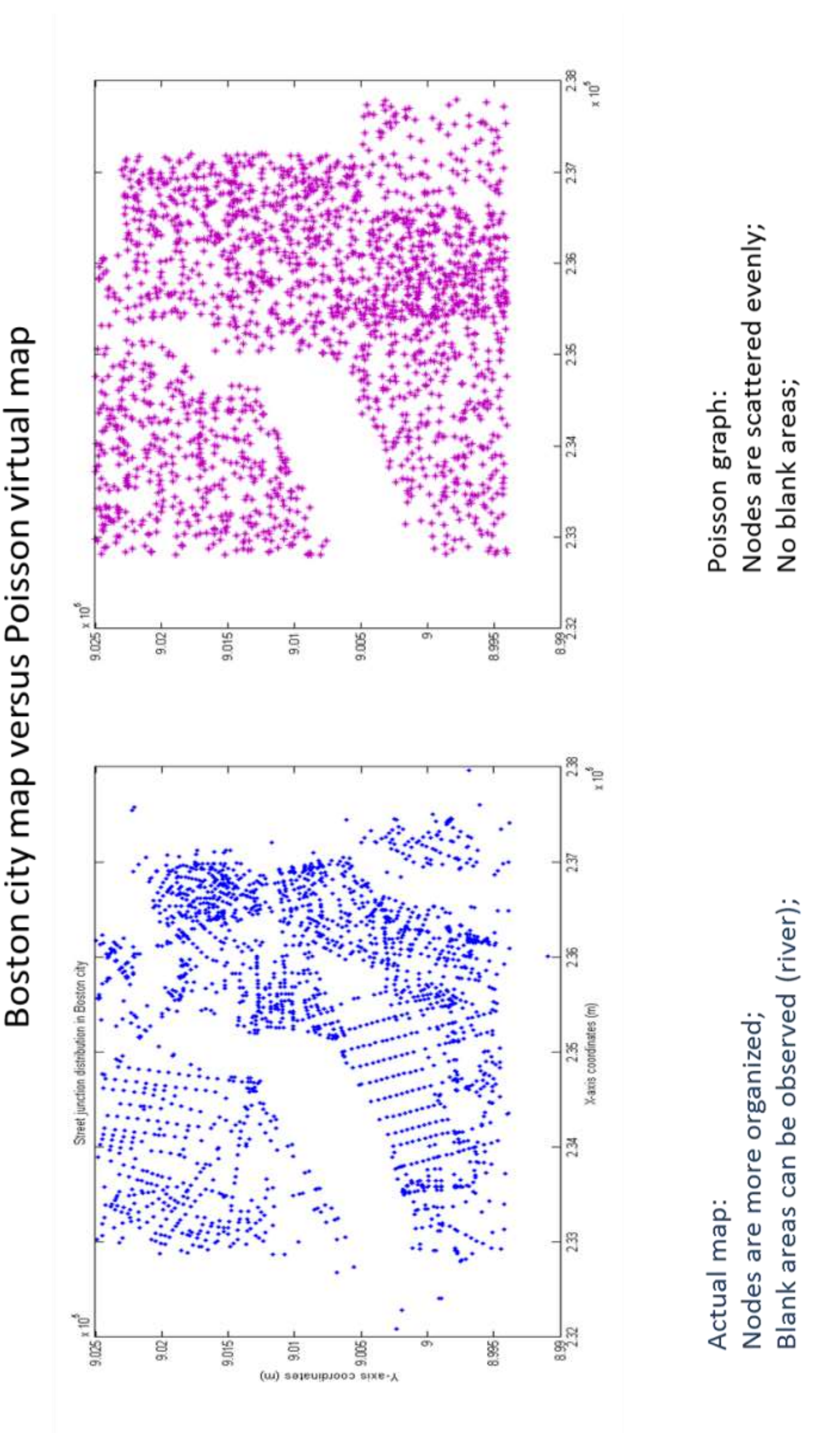


Fig. 3.5 Boston city map and Poisson virtual graph

To summarize, this section has reviewed the process of integrating SG into the Boston city map. In order to apply SG to any geographic object, a survey of its intensity measure must be executed in advance. The following measurements are frequently used in academic research:

- Street junction intensity
- Subway, coach station intensity
- Street length distribution
- Residence density

Different types of intensity measure could result in totally different Poisson point images. For example, if a researcher wants to investigate the public transportation situation of Boston city, he (or she) may choose a coach station number as an intensity measure, and divide Boston city according to the station density distribution. Another situation is a demography analyst, who wants to check the relationship of population density to the signal transmission noise level. His (or her) SG model obviously should choose residence density as its intensity measure.

Another point to be mentioned is that SG is highly suitable for urban area investigation, as city street distribution is quite static and uniformly distributed. Observations on city maps can provide the following statements:

- Capitals of many nations, and their financial and political centers are designed in a rigid and uniform style, which is suitable for stochastic geometry applications;
- City centres, dense residence areas are more regular than gardens, famous attractions and suburb areas;

- Rivers, highways and airports will leave big blank areas and result a dis-continuous behavior in density distribution graph;
- Modern cities are more favorable for SG analysis, because their architectures are well planned, divided into functional districts with regular shapes

Table 3.3 clarifies which scenarios are suitable for SG geometry analysis, and which scenarios are not.

Table 3.3 Categories of city scenarios for stochastic geometry applications

	City	**Building**
Recommended	Nation's capitals; Modern cities;	City centre; Dense residence (offices, shopping malls); Subway stations;
Suitable	Well-designed cities and towns;	City suburb; gardens;
Not suitable	Sparsely distributed cities and towns;	Airports; highways; rivers

3.5. CONCLUSION

This chapter has introduced SG, a useful statistical approach for multi-dimensional data structure analysis. In section 3.1, the PPP was introduced: its constant density property is a fundamental theoretical background in this research. The generation of a Poisson point process followed in section 3.2. Then, the application of this process to modern wireless network architectures was considered in section 3.3. The SNR was linked with the Poisson density parameter λ in equation 2.8 after a series of mathematical deductions. This equation is the key point of the research because it states that the signal loss rate is proportional to the exponential of this area's intensity λ, and it is a series of Galton-Watson processes. This foundation was extended to the Boston city scenario in section 3.4 and the contribution here is to choose street junctions as the intensity measure, and divide the whole city into seven areas, each with similar intensity measures. Furthermore, in order to analyze inter-area data loss rates, a Markov chain was implemented in chapter 5. This method is more precise and more suitable for various city scenario models. Finally, the city scenarios that are suitable for SG analysis are summarized.

REFERENCES

3.1 H. J. A. M. Heijmans, Morphological Image Operators. Academic Press, Boston, MA. 1994.

3.2 J. Goutsias and H. J. A. M. Heijmans, Edison. Mathematical Morphology. IOS Press, Amsterdam. 2000.

3.3 J. Serra, Image Analysis and Mathematical Morphology. Academic Press, London. 1982.

3.4 P. Soille, Morphological Image Analysis. Springer-Verlag, Berlin. 1999.

3.5 E. N. Gilbert, Random Plane Networks, SIAM Journal, vol. 9, pp. 533-543, iv, 1961.

3.6 D. Stoyan, W. Kendall and J. Mecke, Stochastic Geometry and Its applications, 2nd Edition, John Wiley and Sons, 1996.

3.7 P. Guttorp and T. L. Thorarinsdottir. What happened to discrete chaos, the Quenouille process, and the sharp Markov property? Some history of stochastic point processes. Int. Stat. Rev. 80, 253-68. 2012.
3.8 J. G. Andrews, F. Baccelli and R. K. Ganti, A Tractable Approach to Coverage and Rate in Cellular Networks, IEEE Trans. On Commun. Vol. 59, no. 11, Nov. 2011.
3.9 M. Haenggi, J. G. Andrews, F. Baccelli, O. Dousse and M. Franceschetti, Stochastic Geometry and Random Graphs for the analysis and design of wireless networks, IEEE. J. Sel. Areas Commun., vol. 8, no. 7, pp. 3498-3509, Jul. 2009.
3.10 X. Yu, M. S. Leeson and E. L. Hines, Urban VANET Performance Optimization, Advances in Information Sciences and Service Sciences (AISS), June 2013.
3.11 W. P. Tay, J. N. Tsitsiklis, and M. Z. Win, On the impact of Node Failures and Unreliable Communications in Dense Sensor Networks, IEEE Trans. on Signal Processing, vol. 56, Issue. 6, June 2008.

CHAPTER 4

SMART ROUTING ALGORITHM

This chapter introduces SRA (Smart Routing Algorithm): a novel routing scheme for VANET optimization. An example is given throughout this chapter to explain how the SRA works. In section 4.1, a triangle formation is firstly presented. This unit consists of three basic sensing units which are placed evenly near a street junction, and serves as a channel multiplexing and de-multiplexing unit within a VANET. In the simulation, a WSN consists of multiple triangle formations scattering across the whole city. Then two data storage structures are introduced: the Bit Error Rate Matrix and the Sensor Number Matrix. The first of these assists bit error rate (BER) optimization, while the second one is used in sensor number optimization. The main functions of SRA are 1) optimizing BER for a point to point wireless connection, so the quality of service will be improved on the link; 2) minimizing sensor numbers used for the particular route, to save the energy spent in communications. In section 4.2, SRA's operation is described in both the static and dynamic cases, respectively. One can observe how SRA utilizes data stored in the two matrices above and optimizes over two directions. In section 4.3, the computational complexity of this algorithm is analyzed. Although SRA occupies more computational resources in its initiation phase, its biology nature guarantees a rapid decrease in resource utilization in later evolutions. In addition, it always keeps a group of a number of candidate populations for user selection. These populations will be treated as backup solutions, in case primary connections are torn down.

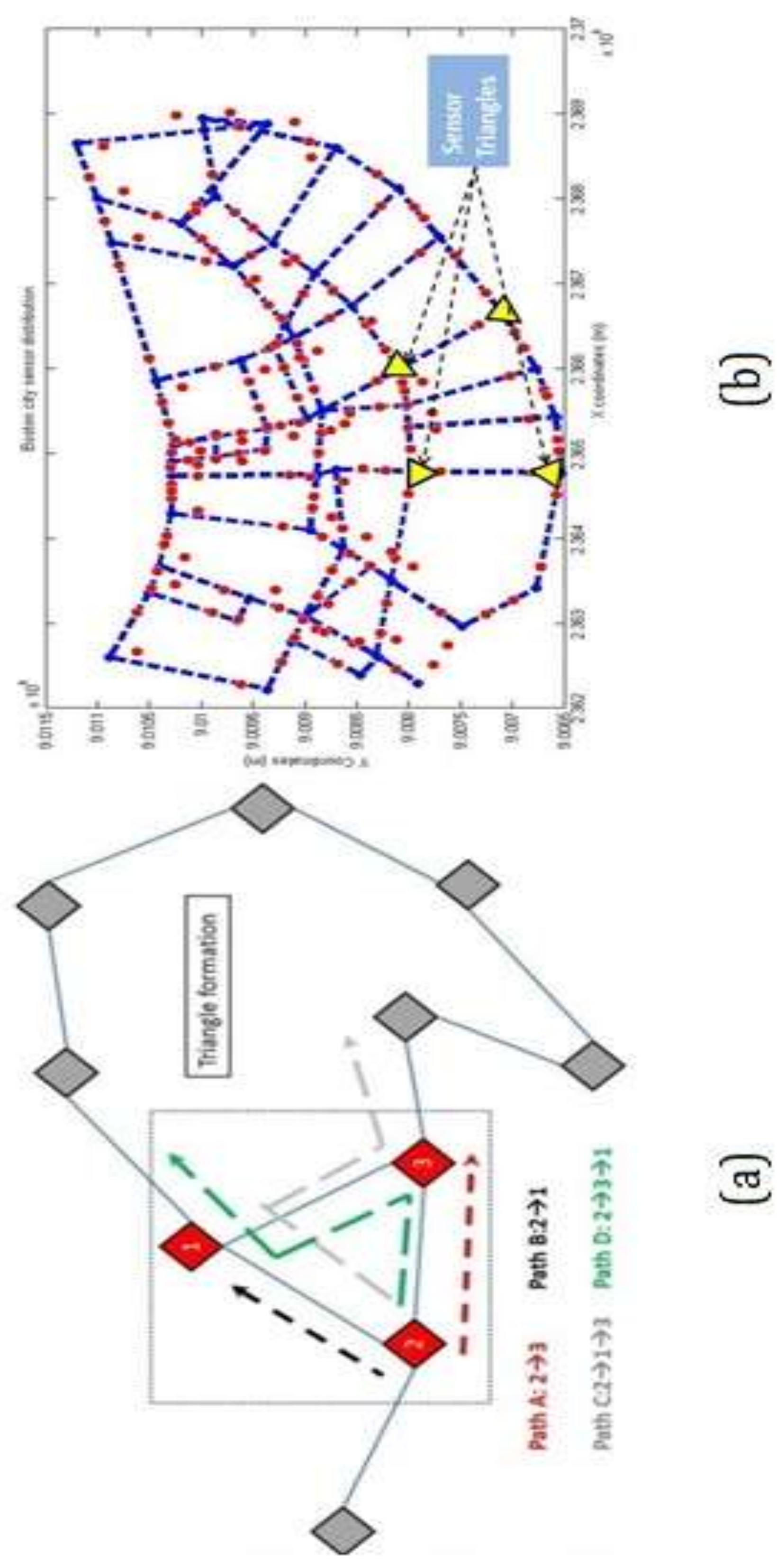


Fig. 4.1 (a) Triangle formation; (b) Boston repeater network

In this section triangle formation in VANETs is introduced. A triangle formation is shown in Fig. 4.1 (a) and, as shown in Fig. 4.1 (b), each Boston street junction contains one triangle formation. The role of this triangle is the same as an electric switch in a network routing device: it provides four candidate routes for any signal passing through. As shown in Fig. 4.1 (a), a signal can choose either route A, B, C, or D.

The reason that a triangle formation was chosen instead of a pentagon or octagon shape is because most of the city street junctions are connected to four other junctions. Evidence for this statement is a statistical analysis from Boston city junctions in Fig. 4.2:

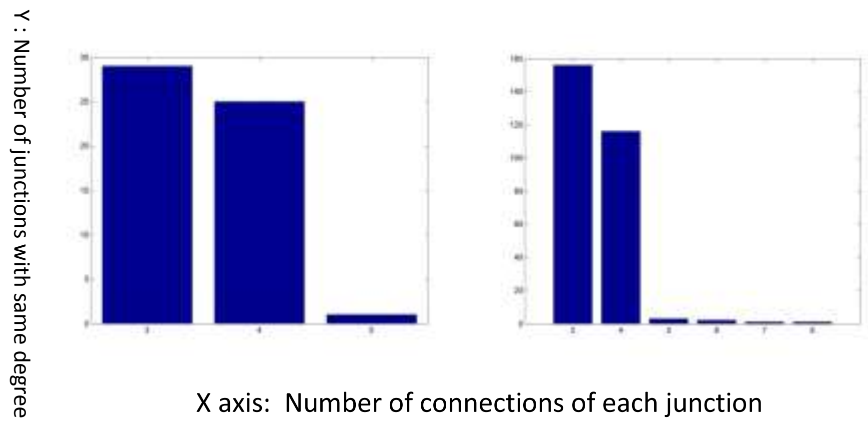


Fig. 4.2 Two PDF bar graphs from Boston city streets

In Fig. 4.2, two bar graphs are presented from statistics of a Boston city digital map. The data it uses originate from a 3-D satellite map of Boston city provided in the MATLAB mapping toolbox. The *x*-coordinates are the number of connections of each junction, and the *y*-coordinates are number of junctions which have the same number of connections. It can be clearly seen that street junctions with degree 3 and 4 dominate the whole population. This is clear evidence that most streets of Boston city are connected to two or three peers.

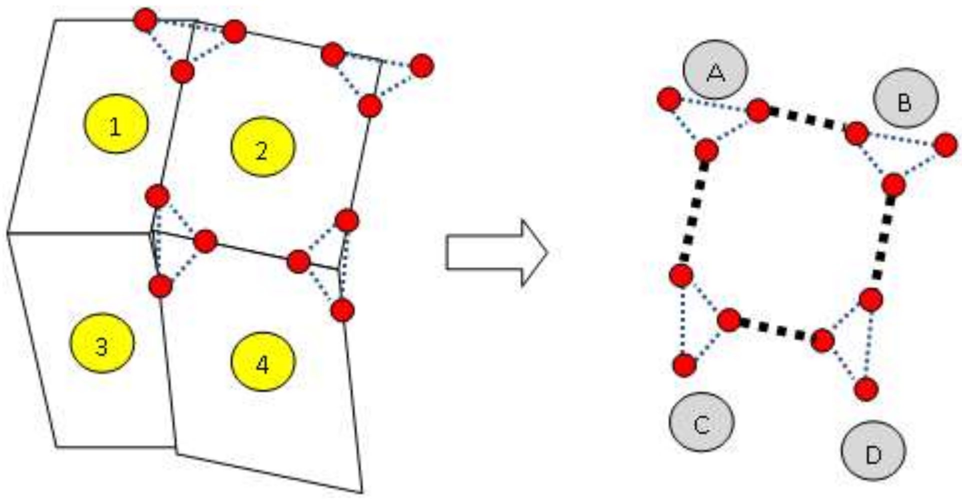
Thus, a triangle formation is sufficient to cover an area linking with 2 or 3 streets. If a pentagon formation is used in this junction, one or more repeaters will need to be placed in the centre of this area, which could result in resource redundancy and unnecessary noise.

As is observed from Fig. 4.1 (b), triangle formations are placed around Boston city. They should be installed 3 to 5 metres high to avoid blockage by pedestrians. Once installation finishes, the real-time fading effects can be verified by sending echo code from one triangle to another. By recoding the signal echo duration and strength, real-time delay information and signal fading level may be obtained. In this simulation, the fading coefficient was set to 10^{-4} per 10 metres. This value is extracted from an empirical path loss model verified in research [4.1] and [4.2]. Since repeaters were placed densely and along the streets, the interference between them was ignored because each repeater's signal coverage can only reach its closet neighbour.

Although setting up triangle formations on signal transmission paths will bring delay to a communication itself, there are clear advantages:

1. It will provide a Line of Sight (LOS) transmission path for wireless signals
2. It will provide alternative routes for wireless communication
3. It will provide a cost-effective supplement to Ad-hoc networks

From Figure 4.3, an urban area with four street grids (here given the name *zones*) is shown. The zones are numbered exactly as data matrices (rows first). Then, four triangles are attached for each zone: the positions are preset as upper left corners in this example. Then the zone map is transformed into the triangle map, as shown on the right side of Figure 4.3. It may be seen that there are two types of connections between members of triangles: one is marked as narrow, blue lines and the other in bold black lines. The first one is *inner-triangle connections*, which have a fixed length (10 metres) and a fixed bit error rate (10^{-4}). The second one is *inter-triangle connections*, whose lengths are different from each other, and their bit error rates depend on the distance transmitted.



(a) (b)

Fig. 4.3 Triangle map formation: (a) urban zones; (b) triangle formation

4.1.1 BER MATRIX

Table 4.1 Inner-triangle data entry

<table>
<tr><td>0</td><td>10^{-4}</td><td>10^{-4}</td><td rowspan="3">Zone 2</td><td rowspan="3">Zone 3</td><td rowspan="3">Zone 4</td></tr>
<tr><td>10^{-4}</td><td>0</td><td>10^{-4}</td></tr>
<tr><td>10^{-4}</td><td>10^{-4}</td><td>0</td></tr>
<tr><td colspan="3">Zone 2</td><td></td><td></td><td></td></tr>
<tr><td colspan="3">Zone 3</td><td></td><td></td><td></td></tr>
<tr><td colspan="3">Zone 4</td><td></td><td></td><td></td></tr>
</table>

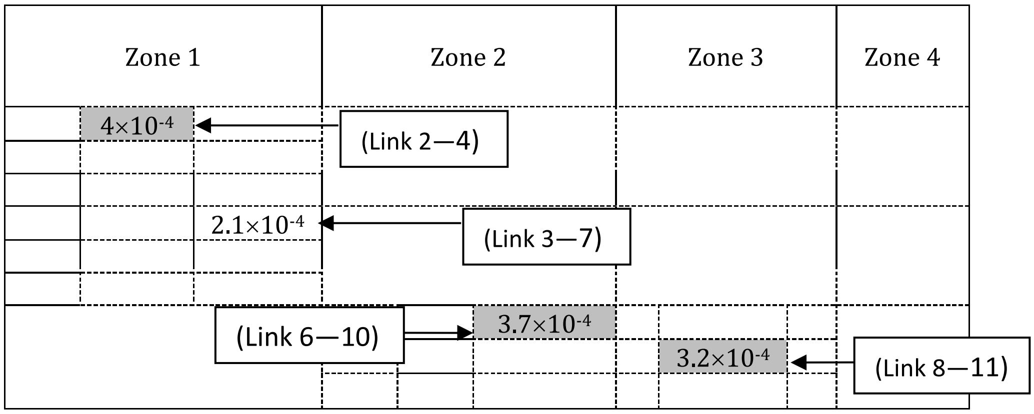
In order to find an optimal route, it is necessary to generate a triangle BER matrix, which is accomplished in two steps:

- Inner-triangle data entry
- Inter-triangle data entry

Step 1: inner-triangle data entry is shown in table 4.1, where the shadowed area is filled with data in a 12×12 BER matrix. This generates a sparse matrix with all inner-triangle connections marked with a preset value (10^{-4}) filled in. It should be noted that zone 1 is anonymous as triangle A—the zone in the digital map, can be named, and so can a triangle in a sensor network architecture, and they can interchange with each other. Recalling the image in Fig. 4.1 (a), the sequence of sensors placed inside a triangle formation is from up-left to down-right. Then, the value of M_{12} represents the BER that occurred in a connection from the upper-left side to its upper-right or down-left peer in triangle A (zone 1).

Step 2: inter-triangle data entry is shown in table 4.2, with the values depending on geographic information from a digital map and the real distance. In this example, a simple linear assumption is made on the relationship between BER and the distance between triangles. Although practically it should be an exponentially-increasing figure, the sensor network can be placed to make the BER increment a linear way—the advantage of a WSN is that sensors are cheap enough to be placed almost everywhere. So in simulation, the algorithm treated BER linearly.

Table 4.2 Inter-triangle data entry



All the data entries in table 4.2 are inter-triangle connection based values. In a practical environment these values shall be reported and updated by repeaters with network broadcasting protocols, but here a linear relationship between BER and street length has been assumed.

In order to avoid algorithm deadlock, all zero data entries were filled with a value representing infinity. In this example, 10×10^{-4} was used instead of infinity to represent zero-data entries. Because infinity is already the maximum that an algorithm can recognize, a population with infinite path length could not be chosen for fitness optimization, as this would have resulted in an insufficient number of populations. The efficiency of an evolutionary program directly relates to the size of its population pool. All in-direct triangles were assigned infinity because they could not be linked with one single connection. So, the search algorithm jumped over them and tried to find other, direct connections.

The BER matrix is initiated when a digital city map is read in by SRA. An index of zones is firstly created, and then street length matrix is read and re-ordered according to the zone index. After the street length matrix is converted into BER values, the BER matrix will read in these values. Because the BER matrix is directly associated with the geographic location of city streets, so it is rarely updated after initiation.

4.1.2 Sensor number matrix

A sensor number matrix records the number of sensors used by an intermediate route between two nodes. In the optimization, the number of sensors will be used as a secondary fitness function. This matrix is created and maintained in exactly the same way as the BER matrix described above. It will include both inner and inter triangle information within it. The final form of a sensor number matrix in this example will be:

Table 4.3 Sensor number matrix

0	1	1									
1	0	1	Zone 2			Zone 3			Zone 4		
1	1	0									
	4										
		2									
					3						
							3				

In simulations based on Boston city, the operations needed to extract map information into these two matrices are listed below:

1. The MATLAB mapping toolbox was used to open the digital image source file 'Boston_city_blue.tiff'; then it read in the data structure containing street information; the output is shown in Fig. 4.4. As observed, this data contained information of street starting point and destination in cell 'X' and 'Y', and it had street length in variable 'LENGTH'. The whole Boston city map contains 3192 streets, so the point matrix has dimensions of 3192×4, and the length matrix is of dimensions 3192×1.

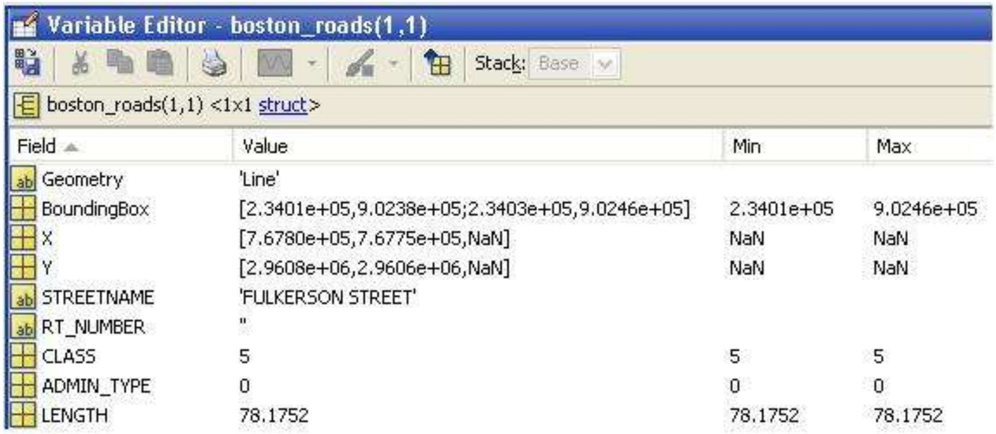


Fig. 4.4 Data structure of Boston city streets

After the data requested was imported into local storage, it was necessary to rearrange the sequence of street points. The order of streets was arbitrary: in this simulation, an index from top-left to down-right was adopted. For example, the city centre of Boston city was divided into 34 zones and labeled from 1 to 34 in sequence, as shown in Fig. 4.5. As was shown in the example before, each zone contains one triangle formation, so, in Fig. 4.5, there were 34 triangle formations in total, containing $34 \times 3 = 102$ sensors. Thus, the BER and sensor matrices were of dimension 102×102.

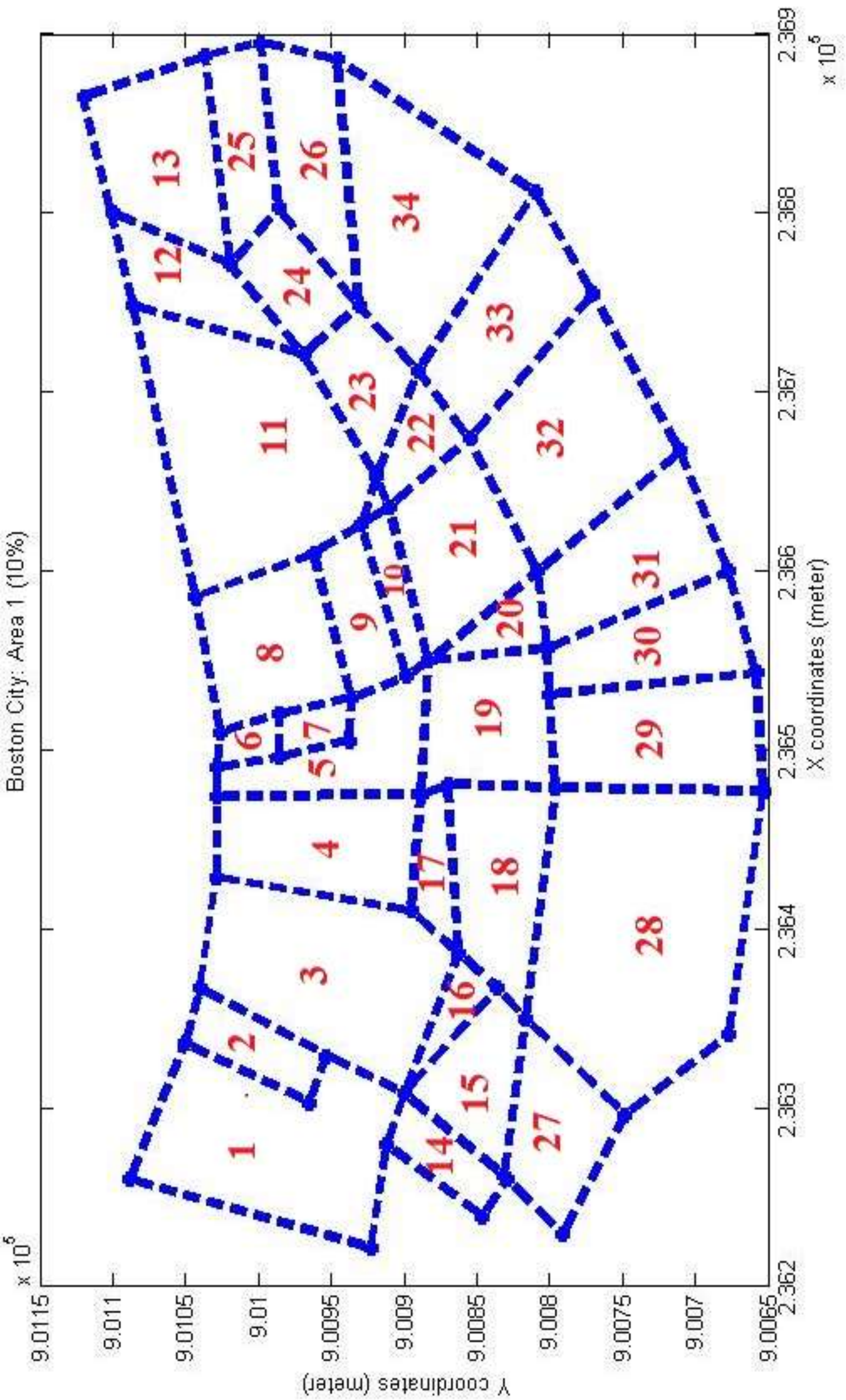


Fig. 4.5 Labeling street zones of Boston city center

2. The initialization of these two matrices was to set all the values contained within them to a starting value representing infinity. Then all the values along the diagonal line of these two matrices were set to zero.
3. To import the values contained in street length matrix into these two matrices, a projection table was needed. This table stored indices of triangles and their corresponding streets. Every street length value had been updated into a cell of these two matrices by referring the cell row as the starting triangle index and the column as the destination index. Once all street length values were imported, these two matrices remained static during algorithm optimization process.

This section has presented how an SRA obtains its information from a digital city map and creates a WSN, based on triangle formations. The transform from geographic objects into two matrices needed some manual operations, such as: labeling city zones; re-indexing streets, and defining the locations of these triangle formations. An example is given on how to setup values contained in the two matrices, followed by an explanation of the Boston city simulation case.

4.2 STREET GRIDS SIMULATION

In this section an SRA application is given, based on a simplified urban scenario consisting of 4 street grids; both static and dynamic cases are treated, followed by analysis of the pros and cons of the optimization process.

4.2.1 STATIC CASE

In the static case, a connection request is oriented from a source node to a destination node, and both nodes are not moving. In this example, a connection request from node 1 to node 12, as shown in Figure 4.6, is considered.

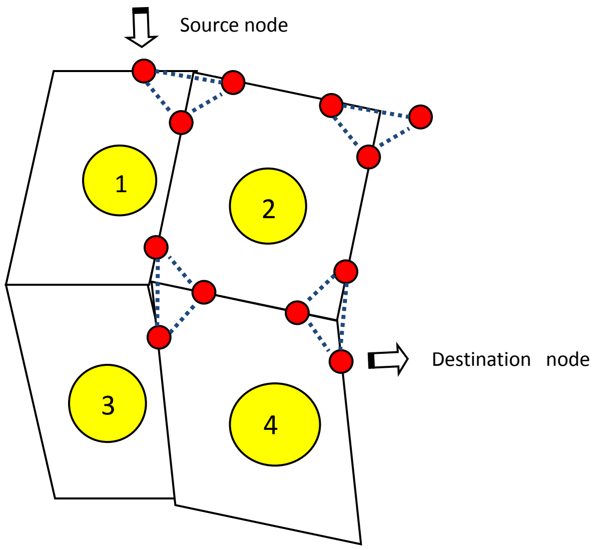


Fig. 4.6 Static connection requests

In the SRA algorithm, each zone is represented by 3 digital bits. In this example, the source node is in zone 1, and destination node is in zone 4, thus the optimal route needs to traverse 4 zones. In this way, a chromosome of $4 \times 3 = 12$ bits will be generated to represent a route. Four chromosomes are randomly chosen as an initial population for optimization. These chromosomes are shown in Table 4.4.

Table 4.4 Initial population: 4 chromosomes

Zone 1	Zone 2	Zone 3	Zone 4
100	001	101	100
010	100	001	100
101	101	011	010
011	001	100	100

Zones 1 and 4 are marked in grey in table 4.4, because they contain source and destination nodes. Thus they have some restrictions in binary expressions. In zone 1, 001 is forbidden because it represents an empty link (neighbouring zones will jump over it). In zone 4, not only is 001 is forbidden, but those bit strings representing links not containing the destination node are also forbidden. Bit strings are randomly generated for each zone, and then the two matrices generated previously will be utilized to calculate initial fitness values.

The fitness calculation process is same for all chromosomes, so chromosome one is taken as an example:

Chromosome 1: 100-001-101-100

Triangle translation: 3rd path—jump over zone—4th path—3rd path

Link generation: 1—2—3 – (no zone 2)—7—9—8—11—10—12

Fitness 1: $\mathrm{BER}_1 = \mathrm{BER}_{1-2-3} + \mathrm{BER}_{3-7} + \mathrm{BER}_{7-9-8} + \mathrm{BER}_{8-11} + \mathrm{BER}_{11-10-12}$

Fitness 2: $\mathrm{Sensor\ no}_1 = \mathrm{Sensor}_{1-2-3} + \mathrm{Sensor}_{3-7} + \mathrm{Sensor}_{7-9-8} +$
$\mathrm{Sensor}_{8-11} + \mathrm{Sensor}_{11-10-12}$

Table 4.5 Fitness values for initial population

Chromosome	Link	Fitness 1: BER	Fitness 2: Sensor number	Possibility to survive
1	1-2-3-7-9 -8-11-10-12	11.3×10^{-4}	14	28.8%
2	1-2-4-5-6 -10-11-12	12.7×10^{-4}	15	25.6%
3	1-3-2-4-6 -5-7-9-11-12	23.2×10^{-4}	17	14.0%
4	1-3-7-8 -9-11-10-12	10.3×10^{-4}	12	31.6%

All the fitness values are shown in table 4.5.

Then, generic GA techniques are performed on these chromosomes:

From table 4.5 (page 78) it may observed that chromosome 1 and 4 are at higher level in the final column: top is a possibility to survive. They will be chosen as a part of new generation. Chromosomes 2 and 3 are worse both in fitness measures (1 and 2). They will be optimized using the generic GA technique: cross over. 4 new generations (shown in Table 4.6) will be created after the crossover process, and the next population will be selected from 2 best existing generations and 4 new generations.

Table 4.6 new populations from crossover operation

Zone 1	Zone 2	Zone 3	Zone 4
100	001	100	100
010	001	001	100
101	100	001	010
011	011	101	100

Table 4.7 shows the fitness values for the newly generated population using a cross over rate of 0.5 and the mutation rate of 0.01.

The next population, from results given in table 4.5 and 4.7, will be chosen to comprise chromosomes 1, 2, 3 and 2'. Compared with the initial population, whose average fitness values are BER = 14.4×10^{-4} and sensor number = 14.5 respectively, the new population reaches an average of 11.5×10^{-4} and 13.8. Thus, the optimization efficiency for a single round is 20.1% and 4.8%.

Table 4.7 Fitness values for initial population

Chromosome	Link	Fitness 1: BER	Fitness 2: Sensor number	Possibility to survive
1’	1-2-3-7-8 -9-11-10-12	$21.3*10^{-4}$	14	19.1%
2’	1-2-(4-6) -10-11-12	$11.7*10^{-4}$	14	34.9%
3’	1-3-2-4-5 -6-10-12	$13.7*10^{-4}$	15	29.7%
4’	1-3-4-6-7 -9-8-11-10-12	$25.2*10^{-4}$	26	16.2%

4.2.2 Dynamic case (source node)

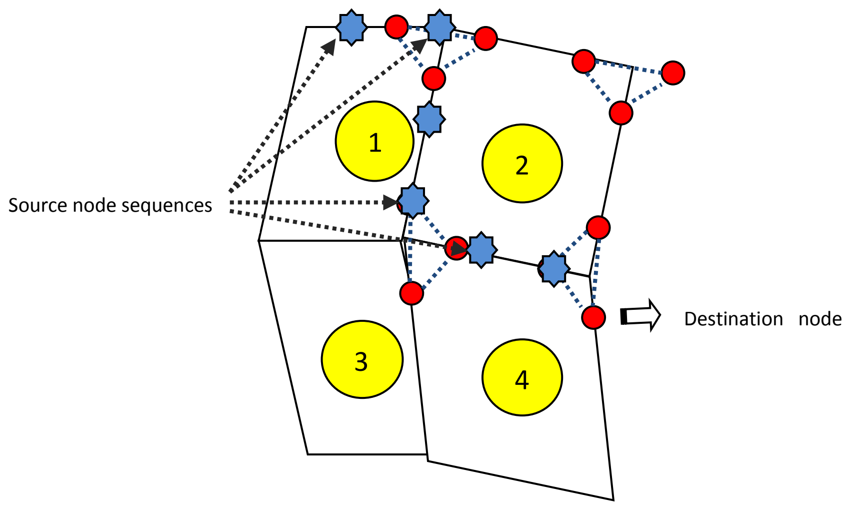


Fig. 4.7 Dynamic source flow chart

In this section, a dynamic node sequence will be given to source node using a popular ad hoc mobility model called RWP. Analysis is presented of how to create chromosomes fitting dynamic scenarios, and how the SRA algorithm performs in this situation.

Thus, in this example, a semi-redundant strategy is proposed to ensure that the optimization process is consistent and efficient all along the node’s movement. Currently the model is a random, no-memory process and thus presents equal possibilities for a node’s next turn in each direction. Thus, a secondary population is kept for each possibility. In this example, the populations will be organized as shown in Figure 4.8.

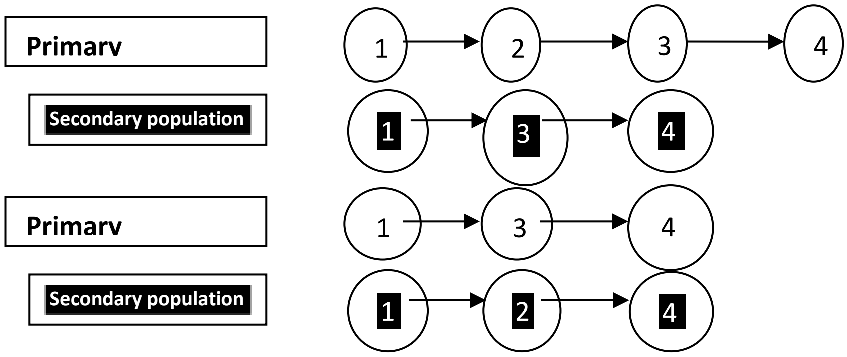


Fig. 4.8 Primary and secondary populations in the dynamic scenario

The primary population is created according to the source node's current position, while the secondary population is created according to source node's predicted position(s). In this example, a random waypoint model [3] [4] is applied: if currently the source node is in zone 1, then it will move to zone 2 and zone 3 with 50% possibilities each. As the primary population already covers zone 2, there is no need to optimize zone 2 twice. Thus its secondary population is chosen as 1-3-4. When source node enters the coverage of zone 3 (triangle 7-8-9), the primary population will update to 1-3-4, and it has been optimized in last round.

A resource redundancy indicator (RRI) will be assigned as a criterion to use the secondary population assisting routing optimization. The RRI records the successful matching rate of secondary populations via primary ones. Its mathematical description is shown in (4.1):

$$\mathrm{RRI} = \frac{\text{Failed secondary populations}}{\text{Total secondary populations}} \tag{4.1}$$

In SRA, secondary populations are used as mutated chromosomes in the optimization process. After optimization, chromosomes surviving to the next round will be examined if satisfying the secondary population requirement. If not enough chromosomes can satisfy the requirements, then the mutation process will be activated to generate more chromosomes for the secondary population.

Table 4.8 Primary and secondary population's evolutions

Source node	1,2	3,4	5,6
Primary zones	1—2—3—4	1—3—4	3—4
Primary populations	011-001-100-100	011-001-100-100	100-100
	100-001-101-100	100-001-101-100	101-100
	010-001-001-100	010-001-101-100	010-011
	010-100-001-100	100-001-100-100	101-011
Secondary zones	1—3—4	1—2—4	3—1—2—4
Secondary populations	011-001-100-100	101-100-001-010	010-101-100-010
	100-001-101-100	010-100-001-100	101-010-100-100

From table 4.8 it can be seen that, when the source node moves from coverage of zone 1 to zone 3, new primary zones are matched by current secondary zones, and so secondary populations are copied into next primary populations. However, the next step of source node from zone 3 to zone 4 causes mutation, because the secondary zone list predicted by the algorithm is 1—2—4, while the actual new primary zone is 3—4. Fortunately, the new primary zone is covered by current primary zone 1—3—4, thus current primary populations can be copied (only the last 6 bits) into new primary populations, while current secondary populations now become redundant.

4.2.3 Multi-objective optimization

In this section a definition of this simulation's search space will be firstly presented. Then follows a definition of the fitness function, which is multi-objective. Based on this information, an estimation of computational complexity of this algorithm is given. Then the efficiency of this algorithm will be given compared with the mainstream Dijkstra algorithm [4.5] [4.6].

The search space associated with this problem is different according to the formation of the algorithm applied to it. For a Dijkstra algorithm, the search space is a *sensor distance matrix*. This distance matrix is the same size of the matrices shown in Table 4.1 and Table 4.2. Thus, given a zone number n_{zn} and a street number n_{st}, the distance matrix is a size of $(3n_{zn})^2$. The worst case complexity for a Dijkstra algorithm is defined as $(3n_{zn})^2+n_{st}$, as it needs to compare $(3n_{zn})2+n_{st}$ data points before reaching the optimal solution. The second matrix has the same configuration as the first one. Thus, the overall computational complexity is defined by (4.2):

$$C_{Dijkstra} = O(2\{[3n_{zn}]^2 + n_{st}\}) \tag{4.2}$$

In the static case, where n_{zn}=4 and n_{st}=4, $C_{Dijkstra}$=O (296). The search space of the SRA algorithm, however, is different from that of the Dijkstra algorithm. The space is a *bit string* space of size $2^{3n_{zn}}$.

Table 4.9 Computational complexity of the SRA algorithm

Procedure	**Operation**	**Computational complexity**
Pre-selection	Choose half best chromosomes from old generation	$1.25n_{zn}\times(5n_{zn}-2)$
Crossover	Crossover half best with the other half to generation new generation	$7.5n_{zn}\times(n_{zn}-1)$
Mutation	Mutate a random chromosome from old generation	$0.15n_{zn}^2$
Fitness calculation	Transform chromosomes into paths and calculate path value using matrix information	$5n_{zn}^3$
Post-selection	Select the other half best chromosomes from new ones	$1.25n_{zn}\times(5n_{zn}-2)$

Since only five possibilities are considered, the number of all possible solutions is restricted to $5^{n_{zn}}$. Suppose the SRA has an initial population of $5n_{zn}$, a crossover possibility of 0.5 and a mutation possibility of 0.01, and the number of evolutions is $5n_{zn}$, then the computational complexity of SRA can be calculated as shown in Table 4.9 (page 83).

In total, the computational complexity of SRA algorithm is thus given by (4.3):

$$C_{SRA} = O(25 * n_{zn}^4 + 100 \times n_{zn}^3 - 62.5 \times n_{zn}^2) \quad (4.3)$$

Here it should be reiterated that C_{SRA} is a quantity representing how many elementary steps are needed to complete the evolution. It has not the same as what conventional search algorithms [4.7] refer to as the "computational complexity [4.8]" which represents the number of basic operations to reach to an optimal point. C_{SRA} is helpful for the estimation of the time cost when the initial population size and evolution round are given. For example, in a real wireless network, when a user is leaving one zone for another zone, the time to calculate a new path is very limited, say 10 seconds. Thus the maximum number of zones is limited to 20, considering elementary operations such as bit-operations and list sorting only to cost 10^{-5} s. This will answer the question of how to choose the initial population size and evolution rounds.

4.3 COMPUTATIONAL COMPLEXITY

Comparing the computational complexities in Table 4.9, fitness calculation is the major contribution with its n_{zn}^3 factor. Thus, how the discovery of an efficient fitness function is essential to guarantee the quality of the SRA algorithm. An efficient fitness function maneuvers the search space efficiently, operates the evolution process stably, and avoids certain pitfalls or deadlocks cleverly. In this section, a multi-objective fitness function is presented, and its efficiency estimated according to its mathematical model.

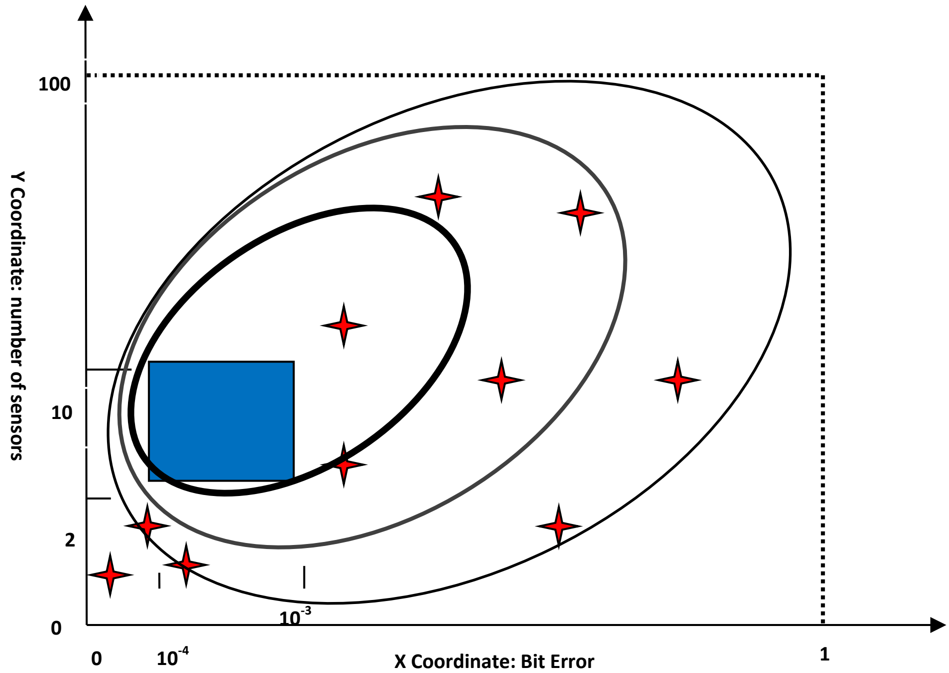


Fig. 4.9 Search space, initial populations, optimal Pareto set

In order to implement a multi-objective optimization algorithm, a search space and a Pareto optimal set need to be defined in advance, as described in Chapter 2. As is shown in Figure 4.9, a search space here is formed by two dimensions: an *x*-coordinate for BER in the the range [0, 1] and a *y*-coordinate for sensor numbers, whose values are in the range [0, 100]. The stars scattered along the space in the figure are initial populations, whose values are randomly distributed. The blue square whose range is [10^{-4}, 10^{-3}] and [2, 10] is the optimal Pareto set—the optimization objective. The task of this fitness function is to generate new populations which can form a new Pareto set whose boundary is closer to the optimal one.

$$Fitness_n = \log_{10}(Fitness_{Sensor,n}) - \frac{1}{\log_{10}(Fitness_{BER,n})} \tag{4.4}$$

In the simulation, the multiple objective optimization fitness function is defined as (4.4). Two initial fitness values gained from new populations are firstly transferred into log space, and then the results are deducted to generate the final result.

The search space defined in the algorithm contains two factors: signal quality indicator (BER) at receiver's side and number of sensors used on the link. An optimization factor space was shown in Figure 4.10 (page 84).

In Fig. 4.10, the fitness function is single objective, so only link signal quality is optimized. As optimization advances, the average link signal quality decreases. However, the *x*-axis, the number of sensors, is not optimized.

In Fig. 4.11 (page 85), the fitness function is a multiple objective one, as defined in Eq. 4.4. The optimization progress evolves as clustering of factor space, which means both the link signal quality and the number of sensors are optimized. From observation of the results in figure 4.11 it may be concluded that this fitness function is optimizing its routing decision to clustering—eliminating both worst and best routing solutions, and leaving those solutions with tolerable signal quality and sensor number usage.

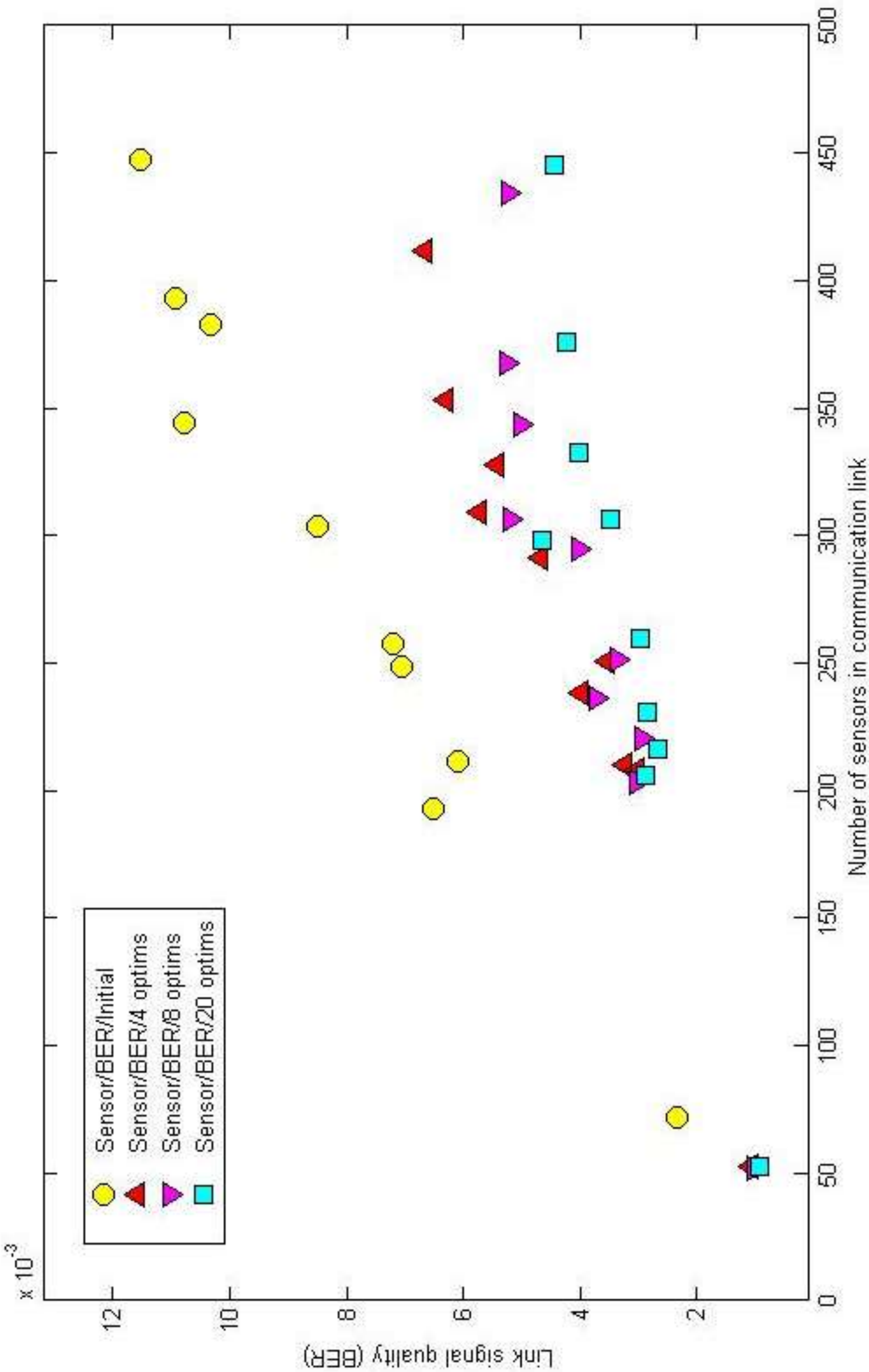


Fig. 4.10. Optimization factor space—single objective optimization

This simulation demonstrates that Eq. 4.4 is able to manage multi-objective optimization tasks by modifying its fitness function.

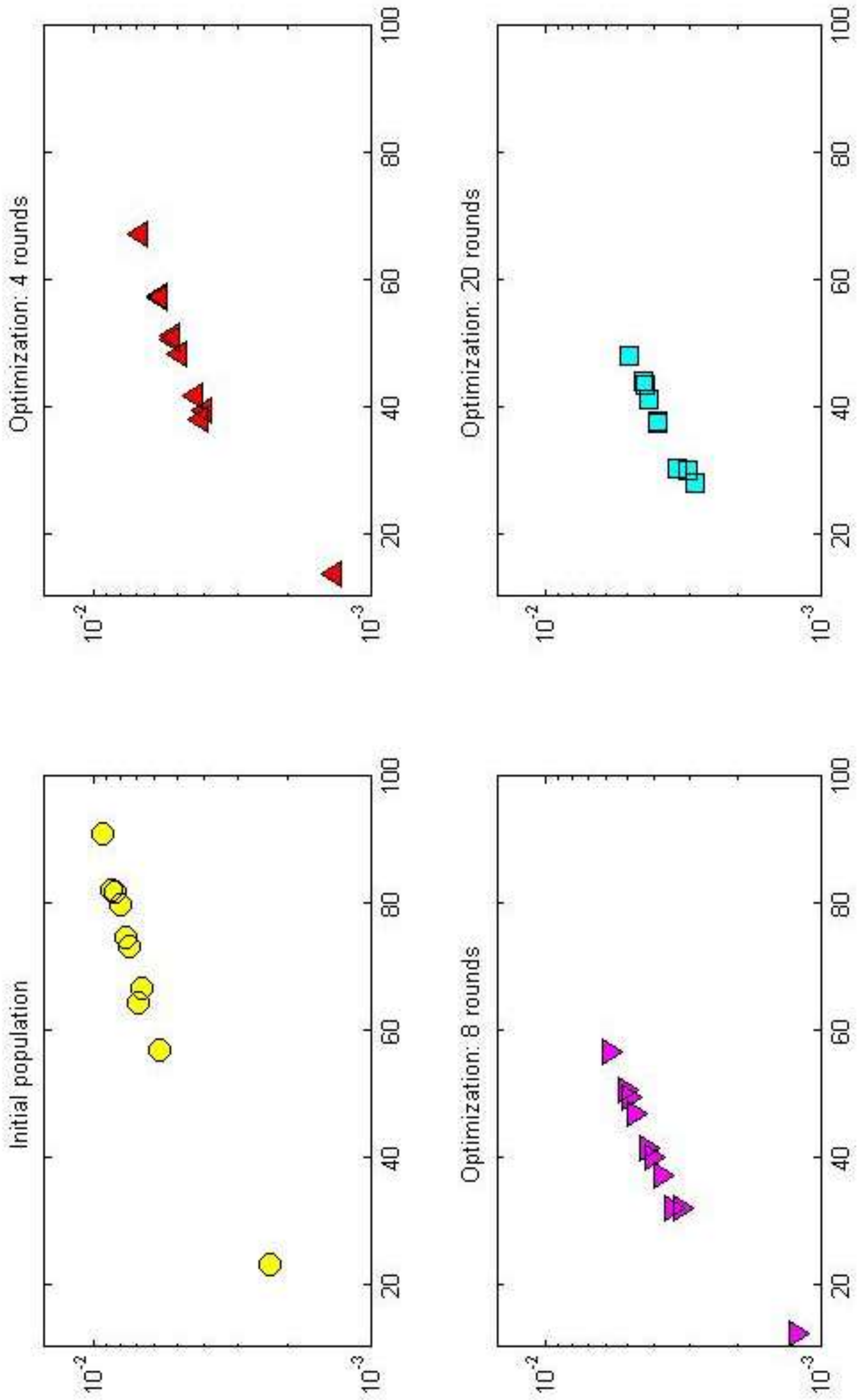


Fig. 4.11 Optimization factor space—multiple objective optimizations

4.4 CONCLUSION

In this chapter, various aspects of SRA have been introduced: initialization, optimization, computational complexity, and capability of multiple objective optimizations. The original contributions include: 1) introducing triangle formation elements, which is key to transforming geographic objects in the city scenario into readable data elements stored in matrices, and is a basis of WSN architecture; 2) proposing a semi-redundant strategy as a variant of EP, so the dynamic network routing case can be solved; 3) analyzing the computational complexity of SRA, highlighting that a major advantage of SRA is that its searching period is independent of the size of data matrices, but relying on the number of bits in its populations. Thus it is possible to control the running period of SRA by carefully selecting its population number and population size; 4) presenting an effective multi-objective fitness function in Eq. 4.4, to assist SRA in multiple objective optimization task. From the analysis and results shown above, one assertion can be made that SRA is an advantageous search and optimizing algorithm in urban network routing optimization.

REFERENCES

4.1. F. Capulli, C. Monti, M. Vari, and F. Mazzenga, Path Loss Models for IEEE 802.11a Wireless Local Area Networks, 3rd International Symposium on ISWCS 06, Wireless Communication Systems, 2006.

4.2. D. Aguayo. Link-level Measurements from an 802.11b Mesh Network. SIGCOMM' 04, Portland, Oregon, USA. 2004.

4.3. C. Bettstetter, G. Resta, and P. Santi, The node distribution of the random waypoint mobility model for wireless ad hoc networks, Mobile Computing, IEEE Transactions, vol. 2, issue. 3, July 2003.

4.4. E. Hyytia, H. Koskinen, P. Lassila, A. Penttinen and J. Virtamo, Random Waypoint Model in Wireless Networks, Networking Laboratory Report, Helsinki, June 16—19, 2005.

4.5. Y. Deng, Y. X. Chen, Y. J. Zhang and S. Mahadevan, Fuzzy Dijkstra algorithm for shortest path problem under uncertain environment, Applied Soft Computing, Vol. 12, Issue 3, pp. 1231—1237, Mar. 2012.

4.6. G. Gallo, S. Pallottino, Shortest path algorithms, Annals of Operations Research, vol. 13, issue 1, pp. 1—79, 1998.

4.7. A. R. Meyers, Computational complexity: theory, techniques and applications. ISBN: 9781461418009. Springer, 2012.

4.8. S. Arora, B. Barak, Computational complexity: a modern approach. ISBN: 9780521424264. Cambridge University Press, 2009.

CHAPTER 5

BOSTON CITY SIMULATION

5.1 INTRODUCTION

A Vehicular Ad Hoc Network (VANET) is an industrial application of mobile Ad Hoc Networks (MANETs). It was introduced firstly in the 20th century on projects with military and city-security backgrounds [5.4] [5.5]. The main differences between a VANET and a MANET are: 1) the scenario of the VANET is mainly focused on real city scenarios whereas research on MANETs normally uses randomly generated (normally distributed) networks; 2) Relay nodes within VANETs are static positions and can be traced with the Global Positioning Service (GPS). On the contrary, the majority of MANET nodes are highly dynamic - only base stations and a limited number of hotspots are static. 3) Vehicles are restricted to move along streets in VANETs, and inter-vehicle communications are greatly impaired by obstacles between them. However in MANETs, users are free to travel in a free space and their communications will only depend on the distances between them, and disruptive noise from other users. Thus traditional MANET models and routing schemes are no longer proper for VANET applications, as concluded by [5.6] and [5.7].

In the past 10 years, a considerable amount of research has been carried on exploring feasible VANET models and designing reliable routing protocols for them. According to published surveys [5.8] and [5.9], current mainstream routing protocols use real-time node information from position-based services (e.g. GPS). Typical routing schemes on VANETs, such as Geographic Source Routing (GSR) [5.10], Greedy Perimeter Coordinator Routing (GPCR) [5.11] and Multi-hop Routing for Urban VANET (MURU) [5.12] all use greedy forward searching techniques and require digital map information from GPS. The main differences between these protocols are the hop selection methods and the vehicle mobility pattern models. As a result, their performance is highly associated with the network conditions.

In order to generate a model applicable to more general VANET circumstances, many researchers have turned to a stochastic geometry approach [5.1] [5.2]. The idea is to use a specifically-designed point process to adapt the situation of the problem (in VANETs it is a real city map). Then attributes of the point process are extracted for further analysis or optimization. In [5.13], the construction of a stochastic model from a real world environment is described in detail. Here Poisson Point Processes (PPPs) are applied on to VANET circumstances.

In order to apply a PPP, it is necessary to take into consideration different node densities of different areas within a target city (as shown in Fig. 5.1). A Multi hop connection process in a VANET considers two geographic situations:

1). All hops are within a same area, where node densities follow a Poisson distribution

2). Hops belong to different areas with different node densities

In the first case, a Galton-Watson branching process (GWP) [5.14] is employed to model a data survival rate (section 5.2 with detailed analysis). The GWP treats multi hop connections as a biological evolution: the node failure rate is assigned to the gene extinction rate. Since all nodes are within an area with the same node density, multi hop connections are transformed into a constantly fading family of genes.

For those multi hop connections crossing over different areas, a Markov Chain [5.15] [5.16] is applied to determine the failure probability of vehicular connectivity (detailed discussion in section 5.3).

In this chapter, a novel Poisson-density and Markov-chain (PDM) process to model a real world VANET circumstance is proposed. This process inherits its mathematical background from [5.1], which constructed a model to predict signal loss rate and base station coverage by treating urban cellular networks as a PPP. Here further analysis of a VANET with a PPP and use of a Markov chain to predict overall signal loss rate across different areas are presented. In order to verify this model's reliability, the research evaluates its performance with a traditional searching algorithm's result, and observes same level of signal loss rate on links with the same spatial distances.

5.2 PROCEDURES

5.2.1 Virtual Poisson graph

In order to apply stochastic geometry on a digital city map, it is necessary to execute the following procedures:

A street junction view of Boston city is in Fig. 5.1 (page 93). It can be divided into 7 sub areas, in order to make each area's junction density evenly distributed:

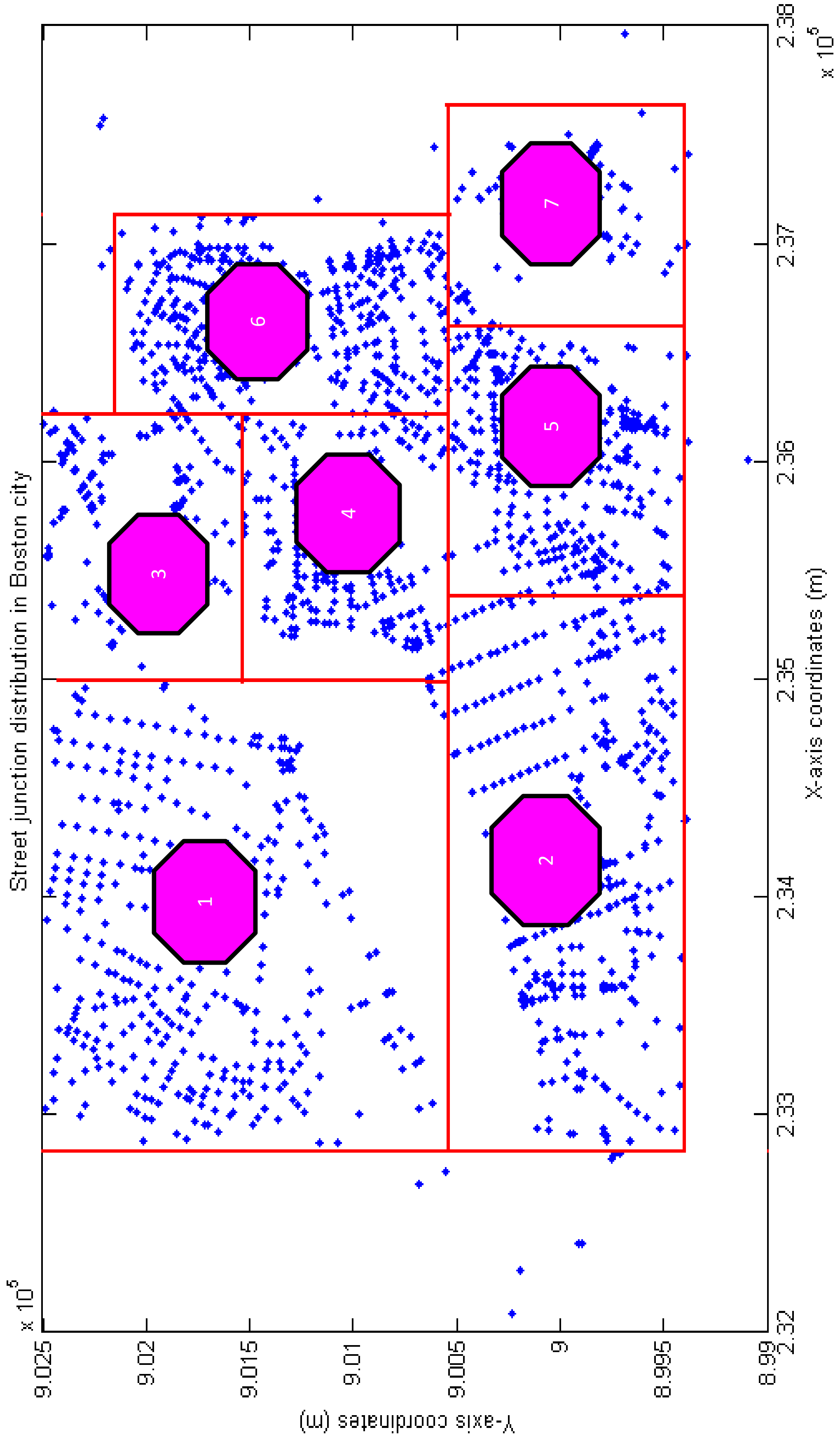


Fig. 5.1 Boston street junctions—7 sub areas

Table 5.1 Boston subarea information

Sub area	Range (km)	Average Density (per 100km^2)
1	[232.8,900.5], [235,9025]	0.8227
2	[232.8,899.4], [235.5,900.5]	1.2323
3	[235,901.5], [236.2,902.5]	1.2250
4	[235,905], [236.2,901.5]	1.8917
5	[235.4,899.4], [236.6,900.5]	2.6061
6	[236.2,900.5], [237.2,902.3]	2.6778
7	[236.6,899.4], [237.8,900.5]	0.7576

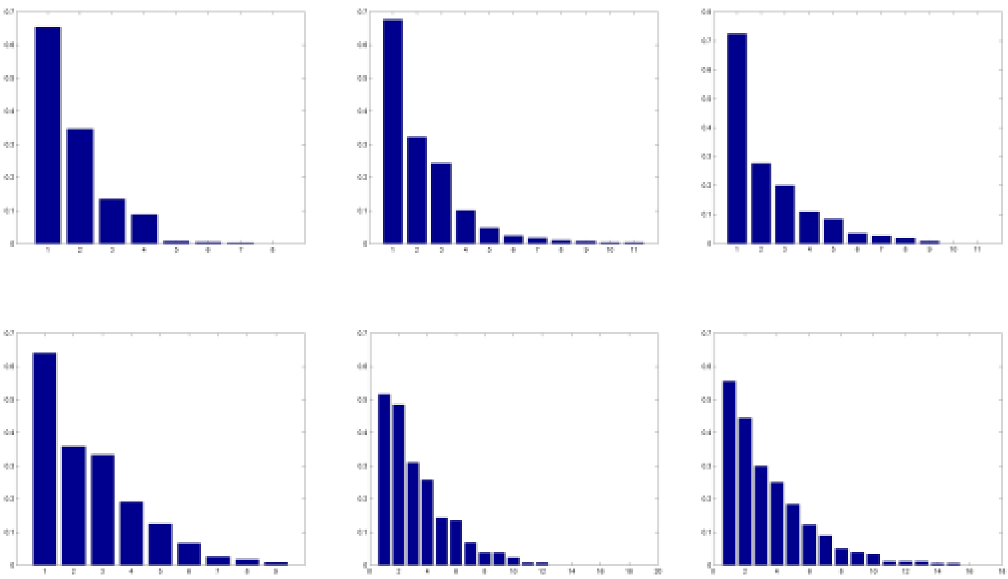
Construct a 2-D spatial Poisson point process using densities' information given in Table 5.1;

Calculate the distribution of street junctions in each sub-area;

Draw the histograms of these sub-areas, and calculate λ;

Histograms of 6 subarea configurations are shown in Fig. 5.1 (page 91).

Boston subarea node



X-coordinates: number of nodes in each cell
Y-coordinates: number of cells with same number of nodes in each subarea (unified to 1)
Graphs on top: subarea 1, 2, 3
Graphs down bottom: subarea 4, 5, 6

Fig. 5.2 node distributions of 6 Boston city subareas

Verify the similarity of this 2-D Poisson map with original city map with Distance methods;

Seven λ values are drawn from Fig. 5.1, listed in table 5.2.

Table 5.2 Values of Poisson model λ in seven subareas

Subarea	1	2	3	4	5	6	7
λ	0.5278	0.4776	0.3793	0.5584	0.9412	0.8	0.32

Seven virtual 2-D Poisson maps are drawn with λ values contained in procedure a); combine them to re-establish the final Poisson map as shown in fig. 5.2;

[1]. Then apply Equation 5.1 to estimate the data loss rate:

$$P_\lambda = \left(\frac{1}{1+\sqrt{\frac{T}{\log\left(\frac{1}{1-p_e}\right)}}\times\arctan\left(\sqrt{\frac{T}{\log\left(\frac{1}{1-p_e}\right)}}\right)} \right) \times e^{\lambda(p_e-1)} \tag{5.1}$$

5.2.2. Data loss rate analysis

In this section, a mathematical theory based mainly on findings in [5.1], is introduced. A vehicle is considered to be in the coverage of a certain VANET node when its received signal-to-noise ratio is higher than a predefined threshold T. Thus, its probability of coverage is defined in Equation 5.2:

$$p_c(T, \lambda, \alpha) \triangleq \mathbb{P}[SINR > T] \tag{5.2}$$

Since here the aim is to explore the data loss rate between multiple hops, this probability of coverage is now a data loss rate probability of a source VANET node in the presence of all other peers. Supposing that a transmitting channel is free of interference (I = 0), the path loss rate is exponential to the distance (general fading), and the fading coefficient equals to 4 (α = 4), an expression of signal channel data loss rate formula results [5.1]:

$$p_c(T, \lambda, 4) = \frac{\pi^{\frac{3}{2}}\lambda}{\sqrt{\frac{T}{SNR}}} \exp\left(\frac{\left(\lambda\,\pi\Re(T)\right)^2}{\frac{4T}{SNR}} \right) Q\left(\frac{\lambda\,\pi\Re(T)}{\sqrt{\frac{2T}{SNR}}} \right)$$

$$\Re(T) = 1 + \sqrt{T}\left(\frac{\pi}{2} - \arctan\left(\frac{1}{\sqrt{T}}\right)\right) \tag{5.3}$$

The Q function in Eq. 5.3 is a standard Gaussian tail probability function. Once the single hop signal loss rate formula is derived, a GWP [5.15] is applied to calculate the multi hop signal loss rate. A complete definition of a GWP and its extinction probability is given in Equation 5.4:

$$X_{n+1} = \sum_{j=1}^{X_n} \varepsilon_j{}^n, X_0 = 1, \lim_{n\to\infty} p(X_n) = 0 \tag{5.4}$$

If ε_j follows a Poisson distribution, a particularly simple relationship can be found between X_n and X_{n+1} as defined in Equation 5.5:

$$X_{n+1} = e^{\lambda(X_n - 1)} \tag{5.5}$$

Now that the multi-hop transmission process has been converted into a GWP with an attached survival probability X_n, a final expression of relationship between Probability of Signal Loss Rate and peer-to-peer transmission error is given in Equation 5.1. In contrary to X_n, this probability is 1-X_n. So it starts from 0 and if $n \to \infty$, will finally approach 1.

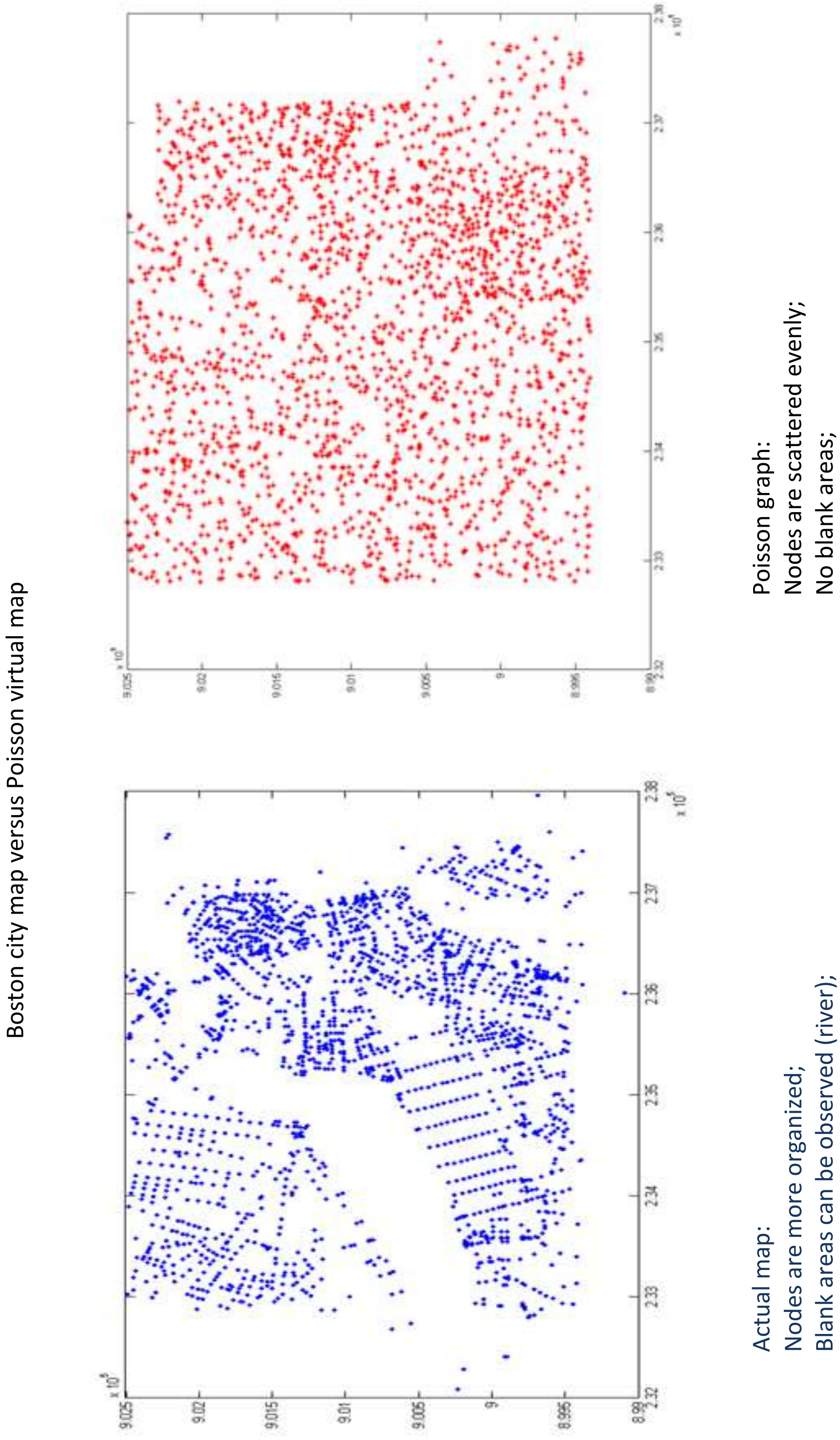


Fig. 5.3. Boston city map versus Poisson virtual map

And it should give results as shown in Fig. 5.4:

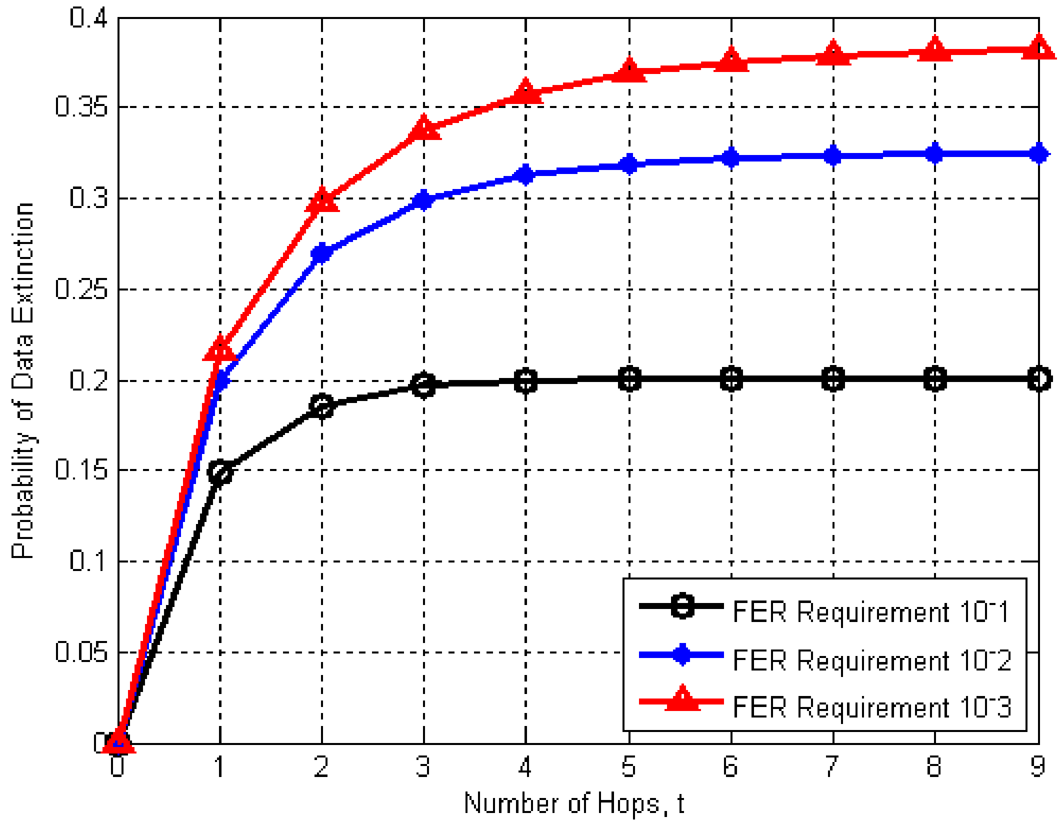


Fig. 5.4 PDF of Data Extension (theoretical)

In order to compare the simulation results from Stochastic Geometry and Evolutionary Programming, the following procedures need to be executed:

a) Sub-area performance comparison
b) Randomly choose 10 points as starting points in each subarea;
c) Decide on 10 destinations, whose geographic distances from starting point are N times proportional to unit hop length (100 meter);
d) Use EP to find the optimum paths from starting point to destinations; EP is restricted to have N number of binary strings;
e) Compare the performance of GA with subplots of Fig. 5.7;

f) Boston city performance comparison;

g) Generate a Markov chain to link different subareas' data loss rates;

h) Use EP to find the optimum paths for these 210 point pairs. EP is restricted to N1 + N2 number of binary strings;

i) Compare the performance of EP to values of transition matrix D in [5.2];

A transition matrix D (7*7) is created to show inter-subarea links;

21 groups of subarea combinations, each group with 10 point pairs are randomly chosen;

In order to guarantee a uniform distribution of these 210 point pairs over geographic distances, some destination points need refinement;

5.2.3. Markov Chain Process

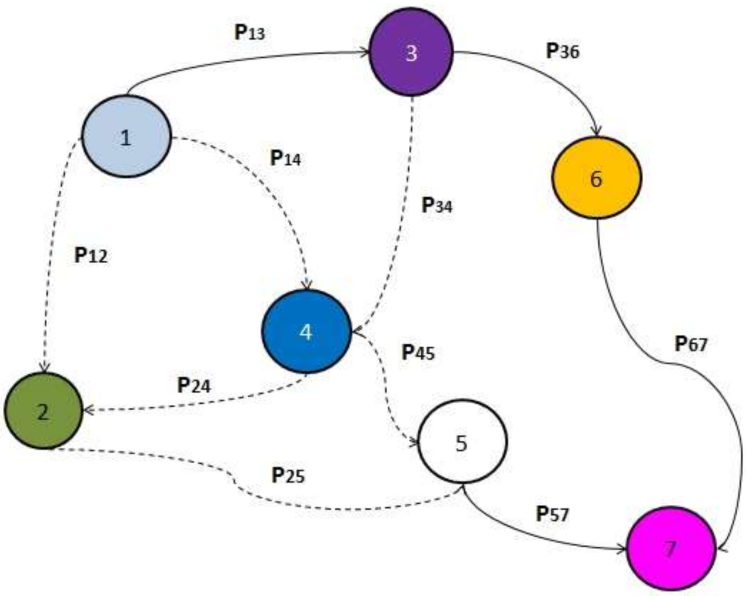


Fig. 5.5 Markov Chain State Flow Graph of the Boston city

An informative graph of Markov Chain formation on 7 Boston areas is presented in Figure 5.5. A complete definition of this Markov Chain is given in a state transition matrix TM. A transition matrix TM from area 1 to 7 is defined in Equation 5.6:

$$\mathrm{TM}(\lambda_1, \lambda_2, \mathrm{n})_{1-7} = \begin{vmatrix} p_{e,1} & p_e^{1-2} & p_e^{1-3} & \dots & p_e^{1-7} \\ p_e^{2-1} & p_{e.2} & p_e^{2-3} & \dots & p_e^{2-7} \end{vmatrix} \; n \in [1, 10] \tag{5.6}$$

Note that TM is a set of multiple transition matrices, according to the value of n (number of hops on the link). The purpose of this algorithm is to find suitable values of $[\, \mathrm{p}_\mathrm{e}^{1-2} \dots \mathrm{p}_\mathrm{e}^{7-6} \,]$ so that a Markov Chain can be established. In this simulation, the possibility of a connection staying within its own area and traveling through the state are set equal to each other. Then $\mathrm{p}_{\mathrm{e},1} \dots \mathrm{p}_{\mathrm{e},7}$ is set to 0.5. The transition matrix TM contains ten variables as is labeled in Figure 5.5. In equation 5.7, seven equations are listed:

$$\textstyle\sum_{j \neq i}^{7} p_{ij} = 0.5\,, i \in [1, 7] \tag{5.7}$$

In order to compute the values of these ten variables, the other three equations are needed. In this simulation, the probability to travel across the area is proportional to the density of this area. Then, the additional equations are listed in equation 5.8:

$$\frac{p_{14}}{\lambda_1} = \frac{p_{24}}{\lambda_2} = \frac{p_{34}}{\lambda_3} = \frac{p_{54}}{\lambda_5} \tag{5.8}$$

Equation 5.8 can be altered because according to the reference area selected by the simulation itself. In table 5.3, several simulation results are shown:

area	p_{12}	p_{13}	p_{14}	p_{24}	p_{25}	p_{34}	p_{36}	p_{45}	p_{57}	p_{67}
1	0.1687	0.134	0.1973	0.1234	0.2080	0.1504	0.2156	0.0289	0.2156	0.2844
2	0.1301	0.1496	0.2203	0.1377	0.2321	0.0408	0.1667	0.1012	0.1667	0.3333
3	0.1278	0.1399	0.2232	0.1476	0.2018	0.1480	0.2121	0.1910	0.2121	0.2879
4	0.1747	0.2133	0.1160	0.0939	0.2377	0.0833	0.1863	0.2068	0.1863	0.2637

Table 5.3 Markov Chain transmit matrix values

After the transition matrix TM is computed, and then the data survival rate can be calculated using data from Figure 5.6 and Table 5.3. Thus the Boston city map transforms into a Markov Chain process, and then a comparison between this strategy and the EP strategy can be achieved (results are shown in Figure 5.7).

5.3. SIMULATION RESULTS

Simulation results are given in Figure 5.6. By observing the curves, it may be concluded:

Data loss rate is counter proportional to the value of λ of each sub-area;

Data loss rate is not linear with the number of relay nodes. When FER = 0.001, the curve is quite exponential, but when FER = 0.1, the curve can be treated as approximately linear;

Comparing with the theoretical curve patterns observed in Figure 5.3, the Boston sub-area curves did not show attributes as heavy-tailed. Instead, they show a constant growth with increased node hops.

Explanations for the three observed differences from Figure 5.6:

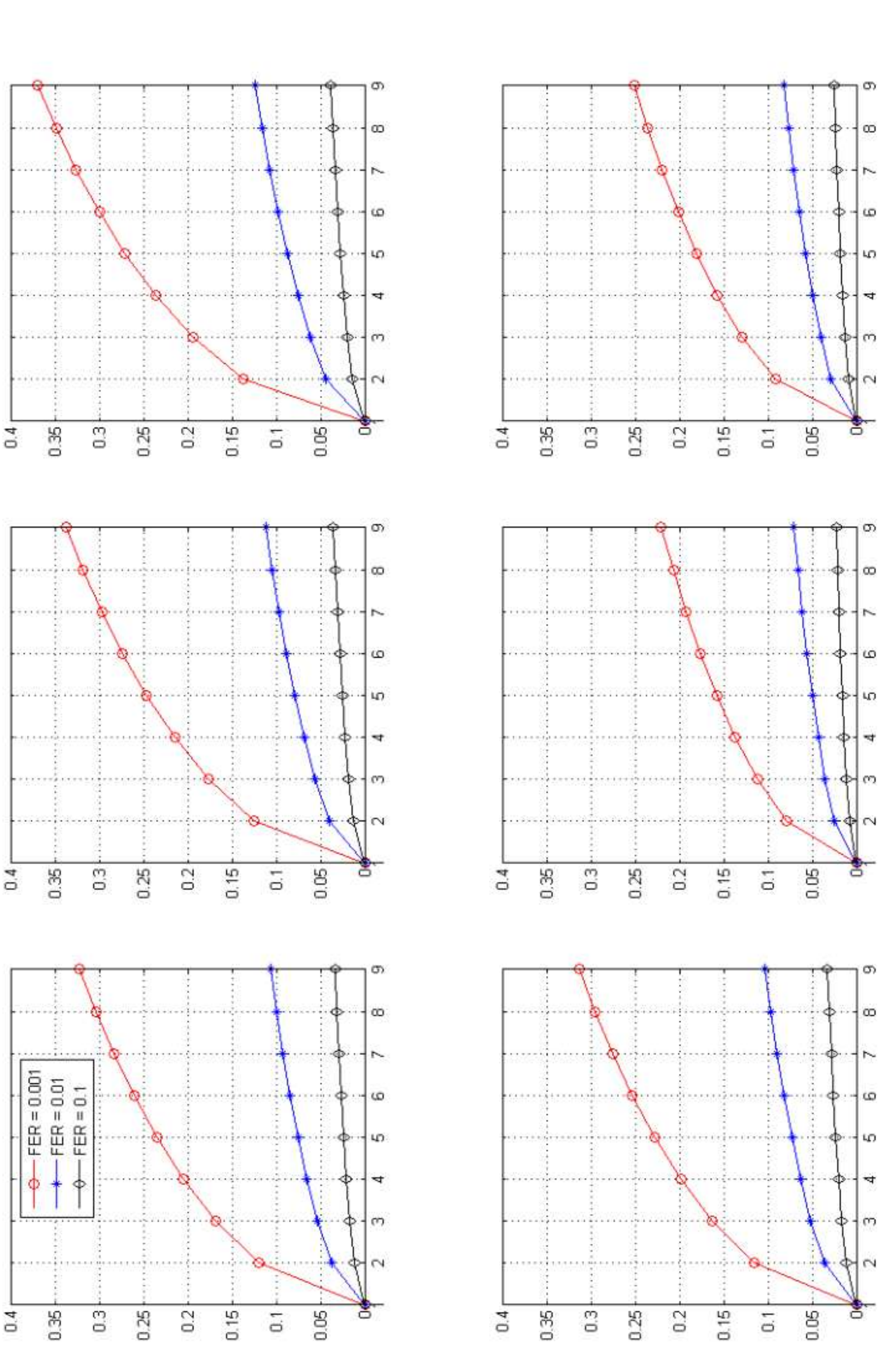


Fig. 5.6 Data loss rate curves on 6 Boston subareas

The parameter, λ, of each sub-area represents the expected density of nodes per unit square. A large λ means a dense sub-area full of street junctions. In the network here, these junctions represent routing equipment to transfer wireless signals. Given two links of same distances but located in different λ-areas, the one with higher λ can benefit from more intermediate routers per unit area. Thus, its data loss rate will be lower compared with its peer who has scarcely few routers to choose from.

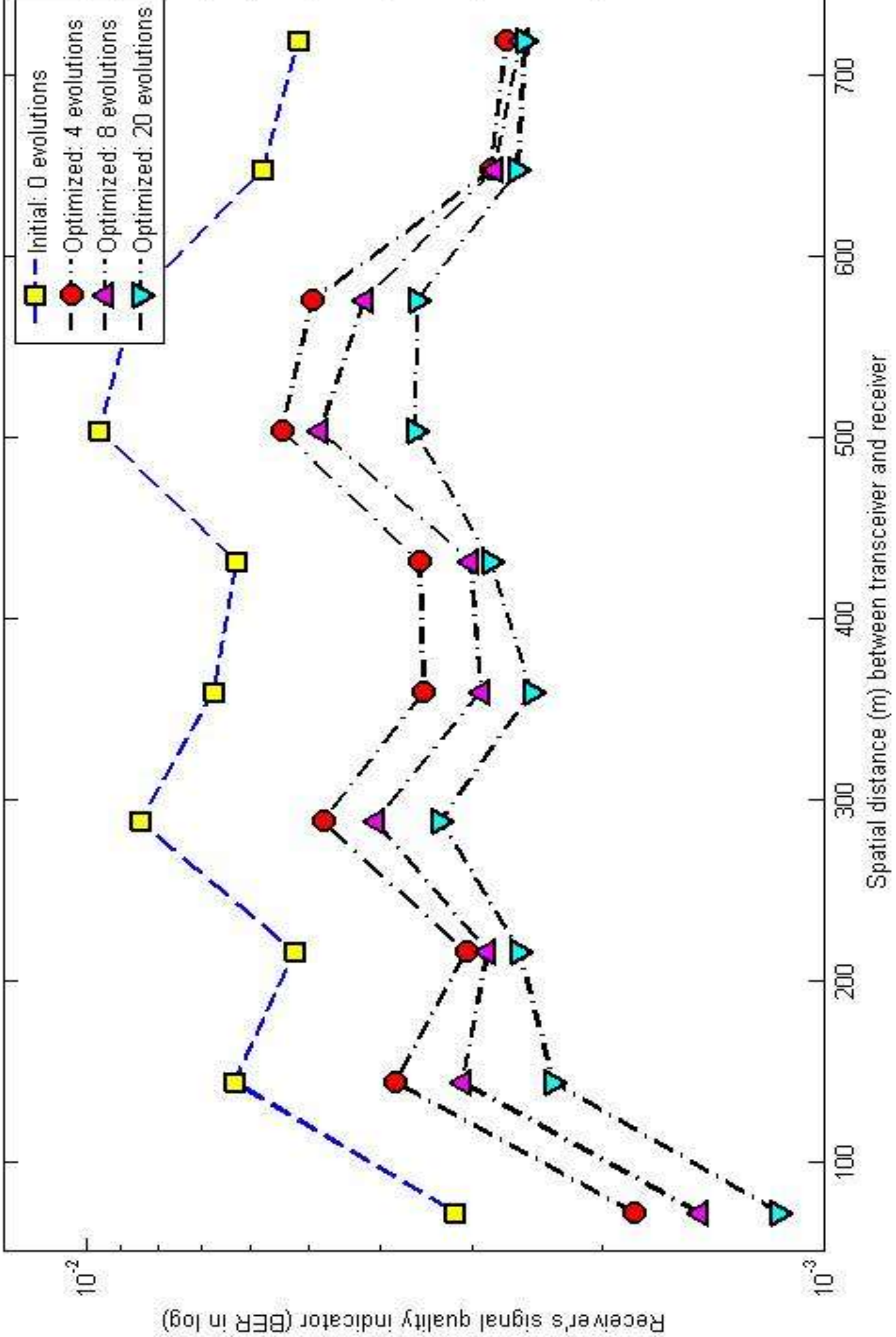
From equation 5.1 one can observe that the data loss rate is proportional to $e^{-\lambda}$ (because p_e <1, so p_e-1 is always negative). So the relationship between DSR (data loss rate) and λ follows on an exponential curve.

In theory, as shown in Fig. 5.3, the data loss rate behaves as heavy-tailed. However, in the simulations here, curves are more linear. Linearity means that signal quality deteriorates smoothly when transmitting over longer distances. In dense city subareas (e.g. number 5, 6), nodes are crowded in such level that signal quality relationship with transmitting distance can be treated linear. In this paper, interferences from neighbouring nodes are not considered. However in the real world, interference cannot be ignored.

The reason why curves in Figure 5.4 do not show heavy-tailed attributes is because the gap between each hop is closer (100 m). However, in Figure 5.3, in order to cover the whole Boston area, hops need to be placed very sparsely. This results in a very heavily tailed curve.

A comparison of EP and SG results is shown in Figure 5.7, from where one can observe the similarity of these two data plots. The x-coordinate represents the geographical distance between a pair of nodes: one acts as a transceiver and the other as receiver. The y-coordinate of the left graph represents signal quality of the data link, while on the right side it means the data loss rate. These two concepts are the same in nature, because the first one evaluates how many data bits are wrongly received by the receiver in percentage, and the second one describes how many data packages are lost in transmission. The bit error rate at the left side is in range [10-2, 10-3] because it is also related to the condition of its communication. However, the data loss rate only relates to the environment and the mathematical model, so it is in range [0, 1].

An obvious difference of these two data curves is the data 'wave' occurring in the first one. This is because random selection of the initial population cannot guarantee the best performance. This disadvantage of the EP strategy can only be compensated by choosing a pool of candidates large enough to include various conditions. In this simulation, the quantity of the initial population is set to 100, which is only 3.13% of the total number of 3192 street junctions. However, as the simulation covers 10 groups, the total population has already covered 31.3% of all possible candidates. Therefore, the opportunity of the population not to include best and worst cases is very rare. However, the dataset from SG simulation is generated by a statistical Poisson process, so it represents the average condition of the data transmission. The stochastic geometry simulation does not taking the physical conditions of the communication channel into account, so its data loss rate purely represents the possibility of a wireless signal to fade away in a specific environment.


Initial: 0 evolutions
Optimized: 4 evolutions
Optimized: 8 evolutions
Optimized: 20 evolutions
Spatial distance (m) between transceiver and receiver
Receiver's signal quality indicator (BER in log)
100
200
300
400
500
600
700
10^{-2}
10^{-3}

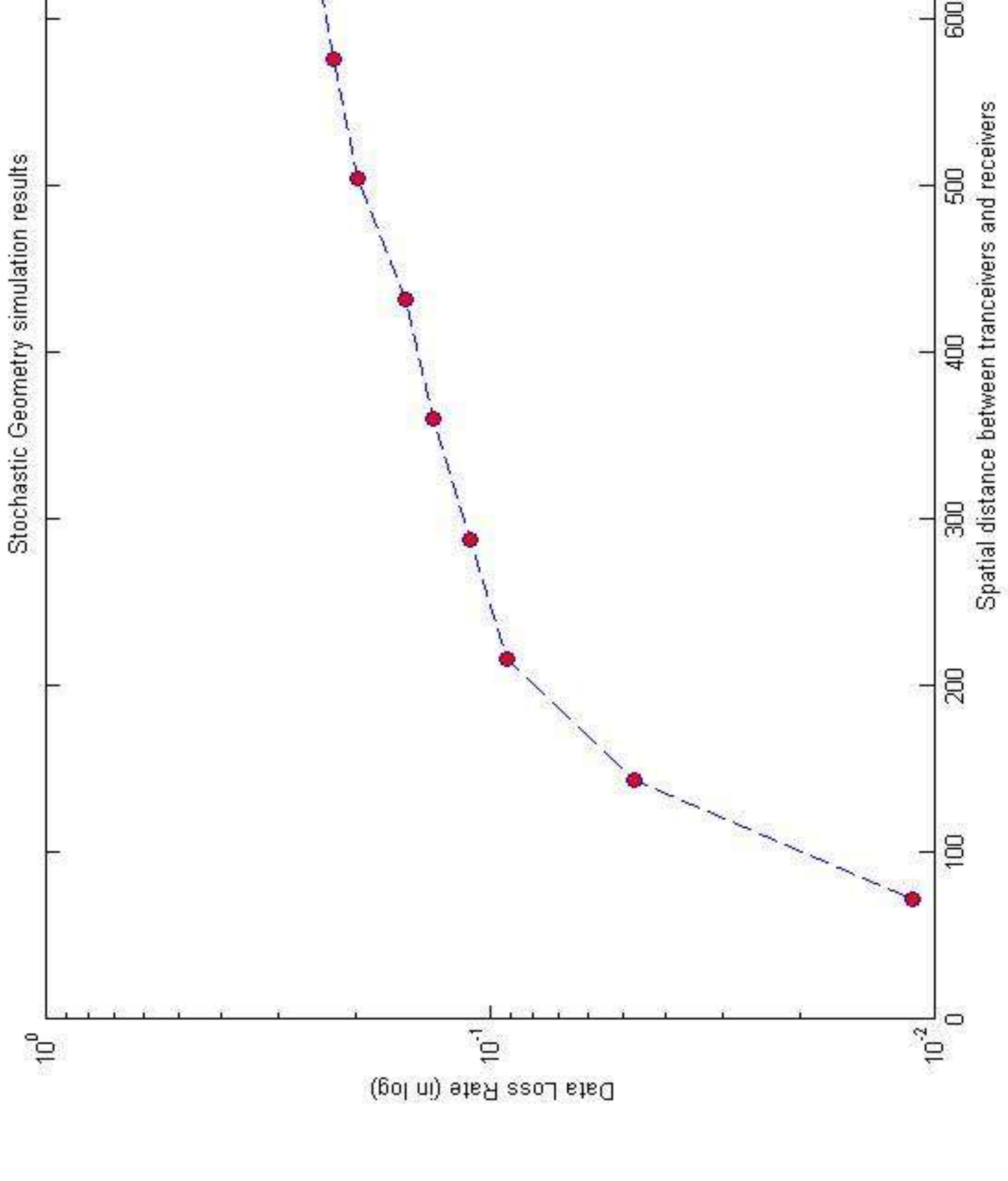


Fig. 5.7 Simulation results of EP (P.106) and SG (P.107)

5.4 CONCLUSIONS

In this chapter, a detailed analysis of a Boston city simulation based on a stochastic geometry technique, was presented. Following a brief literature review, in section 5.2 a virtual Poisson graph was firstly generated with seven subareas, each with its specific density parameter λ. According to a mathematical equation Eq. 5.1, the data loss rate of n numbers of hops could be calculated. In order to compute the data loss rate of the whole Boston city across different sub-areas, a Markov Chain was introduced. After the transition matrix TM specified in Eq. 5.6 is calculated, the whole process was fixed and the data loss rate curve was drawn in Fig. 5.7.

Comparing with the EP strategy, the SG technique was less demanding on geographic information. It only requests statistical information (e.g. junction density, street length distribution). However, EP requests detailed information of the map, such as geographic coordinates, street length, and junction degrees. On the other hand, EP could provide specific results such as bit error rate and shortest route information, while SG could only provide the average value (or the expectation of a specific value). In the optimization process, in order to setup a threshold value for EP, it is possible to apply stochastic geometry on the environment at first to calculate the expected value of signal quality, and then run EP to check how many iterations it requires to reach this value. In this simulation, as the number of street junctions is only 3192, so 100 populations with 20 evolutions can cover the whole candidate pool. In future applications, when the size of data set is much larger, this strategy can be adopted.

REFERENCES

5.1 J. G. Andrews, F. Baccelli and R. K. Ganti, A Tracktable Approach to Coverage and Rate in Cellular Networks, IEEE Trans. On Comms., vol. 59, No. 11, Nov 2011.

5.2 M. Haenggi, J. G. Andrews, F. Baccelli, O. Dousse, and M. Franceschetti, Stochastic Geometry and Random Graphs for Analysis and Design of Wireless Networks, IEEE J. on Selected Areas in Comms., vol. 27, No. 7, Sept 2009.

5.3 X. Yu, M. S. Leeson and E. L. Hines, Urban VANET Performance Optimization, Advances in Information Sciences and Service Sciences (AISS), June 2013.

5.4 D.A. Magder, P. Bosch, et.al, 911-NOW: a network on wheels for emergency response and recovery operations, pp 113-133, Vol. 11, Issue 4: Bell Labs Technical Journal, 2007.

5.5 W. Enkelmann, FleetNet-applications for inter-vehicle communication. Intelligent Vehicles Symposium, 2003. Proceedings. IEEE, pp 162-167. 2003.

5.6 D. Reichardt, et.al, CarTALK2000: safe and comfortable driving based upon inter-vehicle communication. Intelligent Vehicles Symposium, 2002. Proceedings. IEEE, pp 545-550. 2002.

5.7 F. Li and W. Yu, Routing in vehicular ad hoc networks: A survey. Vehicular Technology Magazine, IEEE, vol. 2, no. 2, pp 12-22. 2007.

5.8 Y. Toor, P. Muhlethaler, and A. Laouiti, Vehicle Ad Hoc networks: applications and related technical issues, Communications Surveys & Tutorials, IEEE vol. 10, no. 3, pp 74-88. 2008.

5.9 C.E. Perkins, E.M. Royer, Ad-hoc on-demand distance vector routing, Mobile Computing Systems and Applications, IEEE, pp 90-100. 1999.

5.10 B, KARP, H.T. Kung, GPSR: greedy perimeter stateless routing for wireless networks. In Proc 2000 Mobile computing and networks: pp 243-254. 2000.

5.11 M. Torrent-Moreno et al., Packet forwarding in VANETs, the complete set of results. Technical report 2006-2, ISSN 1432-7864, Germany. 2006.

5.12 S. Y. Tan and H.S. Han, A theory for propagation path-loss characteristic in a city-street grid, IEEE Transactions on Electromagnetic Compatibility, vol. 37, no. 3, pp. 333-342, 1995.

5.13 D. Stoyan, W.S. Kendall and J. Mecke, Stochastic Geometry and its Applications (Second Edition), John Wiley and Sons, ISBN: 0471 950998. 1994.

5.14 W. P. Tay, J. N. Tsitsiklis, and M. Z. Win, On the impact of Node Failures and Unreliable Communications in Dense Sensor Networks, IEEE Trans. on Signal Processing, vol. 56, Issue. 6, June 2008.

5.15 S. M. Abd El-atty, Vehicular Communications Framework for Efficient Multihop Connectivity in AHVN. Vehicular Technology Conference, Fall, 2011.

5.16 R. Shirani, F. Hendssi, A Markov chain model for evaluating performance of Store-Carry-Forward procedure in VANETs. IEEE Communication Systems, ICCS 2008, pp. 593—598.

CHAPTER 6

SMALL WORLD NETWORKS

6.1 LITERATURE REVIEW

The idea of 'small-world' came from a famous social phenomenon exposed by Stanley Milgram in 1967. In his experiment, Milgram tracked chains of acquaintances from two randomly chosen people in the U.S. to see how many intermediate messengers were required to establish a social connection. The feedback from 160 visitors confirmed that the length of the chain was a number between two to ten, with a median of five. This finding resulted in his publication of [6.1] and the famous concept called 'six degrees of separation'. This term represents an extremely simple finding in sociology that a message can be sent to anyone in our social network by delivering it to close friends or relatives for six times, thus it is called 'small world phenomenon'.

After the notion of 'six degrees of separation' was proposed, this model was soon adopted by social scientists and economists and became a standard network diagram in their research. In 1998, Watts and Strogatz proposed their model [6.2] on the small world phenomenon. This model generated a random Erdos-Renyi (ER) graph [6.3] with two small-world properties: betweenness [6.4] and clustering [6.5]. Betweenness is a network property to evaluate the average path length needed to connect any pair of vertices. Clustering is a measurement of network transitivity or the number of node sets contained in a network.

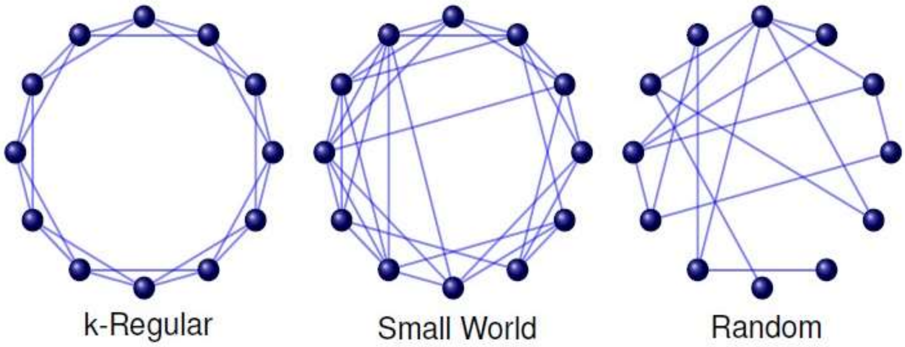


Fig. 6.1 Small world effect diagrams

As shown in Fig. 6.1, the generation of networks by applying the Watt-Strogatz model can be divided into three phases, according to the value of 're-wiring probability' β. This network is a k-tier lattice if β = 0 (shown at the left side graph in Fig. 6.1); it will become a network with small-world probabilities if 0<β<1, as shown in the center graph in Fig. 6.1; and will be a completely random graph if β=1.

Small-world network properties

The following section introduces two key small-world network properties, namely average length and clustering coefficient.

Average length

Given a network model of N number of nodes, the mean node degree K, and the re-wiring probability β, a small world network can be generated with the Watts-Strogatz model. The average path length is defined as equation 6.1:

$$\begin{cases} l(\beta) = \frac{N}{2K}, for\ \beta = 0 \\ l(\beta) = \frac{\ln(N)}{\ln(K)}, for\ \beta = 1 \end{cases} \tag{6.1}$$

In equation 6.1, l(0) represents the path length when β = 0, and l(1) represents the path length when β = 1. However, in the range of (0, 1) no formula can be given. The average path length falls quickly when β increases, and finally reaches its limit.

Clustering coefficient

Given a network model of N number of nodes, the mean node degree K, and the re-wiring probability β, a small world network can be generated with the Watts-Strogatz model. The clustering coefficient is defined as equation 6.2:

$$\begin{cases} C(0) = \frac{3(K-2)}{4(K-1)} \\ C(\beta) = \frac{3 \times number\ of\ triangles}{number\ of\ connected\ triples} \\ C(1) = \frac{K}{N} \end{cases} \tag{6.2}$$

In equation 6.2, C(0)) represents the clustering coefficient when β = 0, and C(1) represents the clustering coefficient when β = 1. C(β) can be calculated using the formula provided in Eq. 6.2.

From the introduction of the small-world model and its characteristics, one conclusion can be made that the re-wiring probability, β, is critical in generating a small-world model and analyzing its properties. When β = 0, the small-world model becomes a k-regular model, with average path length equaling $N/2K$ and clustering coefficient equaling $3(K-2)/4(K-1)$. When β = 1, this model becomes a random network model, with average path length equaling $\ln(N)/\ln(K)$ and clustering coefficient equal to K/N. The small-world model requires that $0 < \beta < 1$: when β increases, the network becomes denser and more clustering, so the average path length will decrease and the clustering coefficient will increase rapidly. In this simulation, the average path length of the network directly associates with signal quality (bit error rate), while the clustering coefficient relates to the energy cost of the network. In order to find an optimal topology of this network, a number of network models with different re-wiring probability β are generated in section 6.3, and then the simulation results are compared with the results provided by previous EP and SG simulations.

6.2 SMALL WORLD PROPERTIES AND GENERATION

Given the desired number of nodes N, the mean degree K, and a re-wiring parameter β, satisfying:

$$\begin{cases} \beta \in [0,1]; \\ K \in [N, \ln(N)]; \end{cases}$$

A Watts-Strogatz model will construct an undirected graph with N nodes and NK/2 nodes, by execution of the following operations:

1. Create a regular ring lattice with N nodes, each connected to K neighbours, evenly distributed on each side.
2. For every node connected to node i, take its edge (ni, nj) with i<j, and rewire node i with probability β. The rewiring procedure:
3. Choose a destination node k from a population;
4. Duplicate node k to node j;

5. In order to avoid dead loops, node k shall be chosen from a number of candidates which satisfy the requirement that node i, j, k is not in a closed loop.

6.3 BOSTON CITY SIMULATION

In order to compare the performance of small-world model with previous simulations using EP and SG techniques, the simulation is based on Boston city map as well, Figure 5.1 from chapter 5 is reproduced here for reference:

The procedures needed to transform this digital map into a small-world network are:

1. Divide the digital map into sub-areas and extract their information for simulation;
2. Choose a set of points as powerful sensors, which are called sink sensors in WSN or base stations in Wireless networks;
3. Assume that these points can communicate directly with their CLOSEST peers, so they organize a network of their own;
4. To assign this network with small-world attributes, it is necessary to choose a re-wiring probability β ranging from value (0, 1), then following the procedures regulated in section 6.2, these points can create a small-world network;
5. previous EP and SG models, two factors will be calculated:
    - 5.1. the average path length to traverse the network;
    - 5.2. the average energy cost to traverse the network;

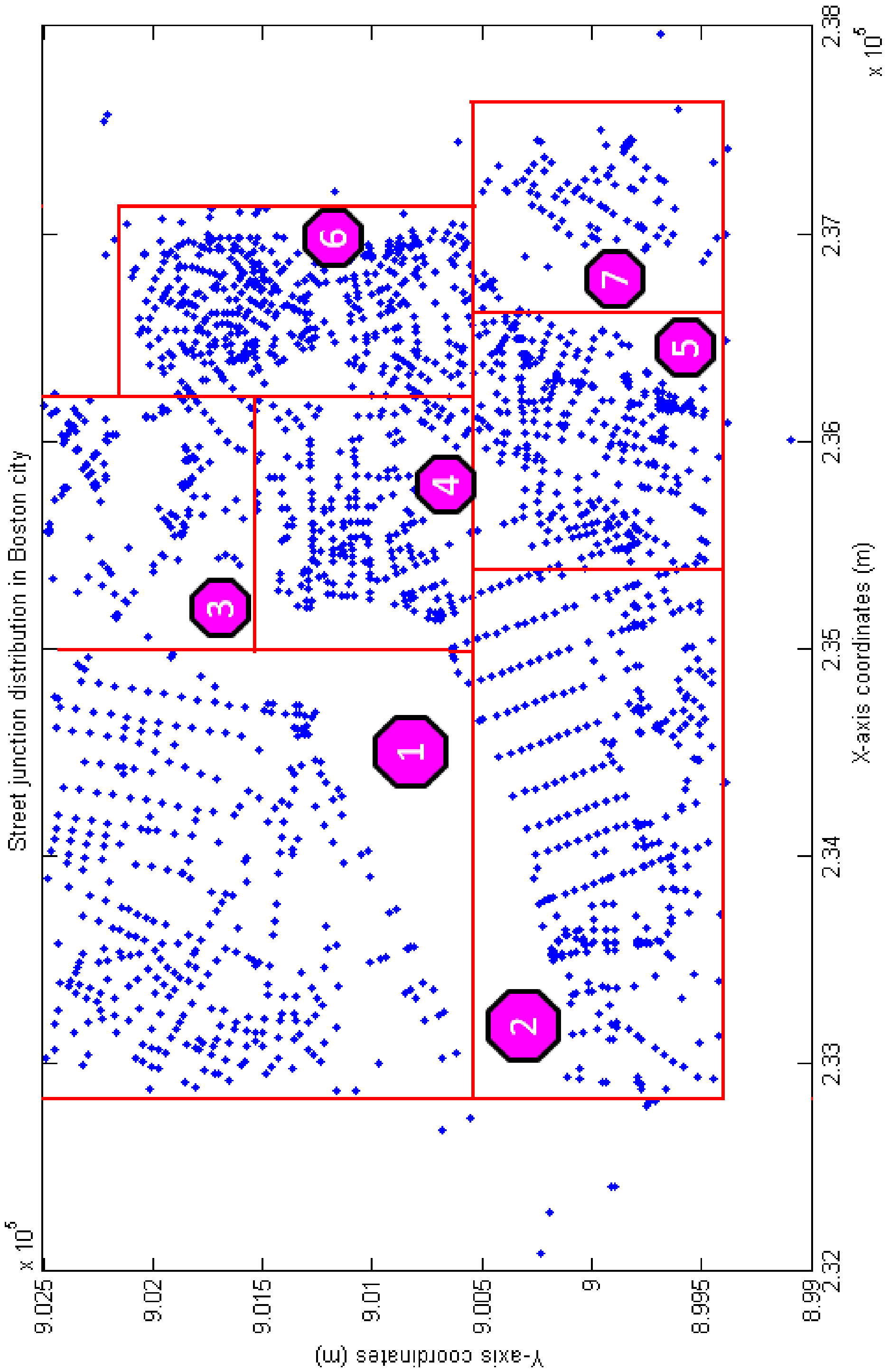


Figure 6.2 Boston city junction map for small-world simulation

6. 100 random pairs of nodes are chosen for this performance test. In EP simulation, 20 epochs of optimizations are set; in SG, these values are given based on node density λ contained in Table 5.2 and equation 5.1. In small-world simulation, depending on the re-wiring probability β, the values need to be calculated accordingly.

In this simulation, β is set to three values: 0.1, 0.5 and 0.8. The number of sink sensors (base stations) is set to 51, which is 2.51% of the number of Boston street junctions (2031 junctions in total). The communicating range of these sink sensors is set to 1.5 kilometers, so they can communicate with their closet peers. A diagram of these sink sensors' distribution is shown in Figure 6.2. Figure 6.3, 6.4 and 6.5 show the map information with re-wiring probability β = 0.01, 0.1, and 0.6 respectively.

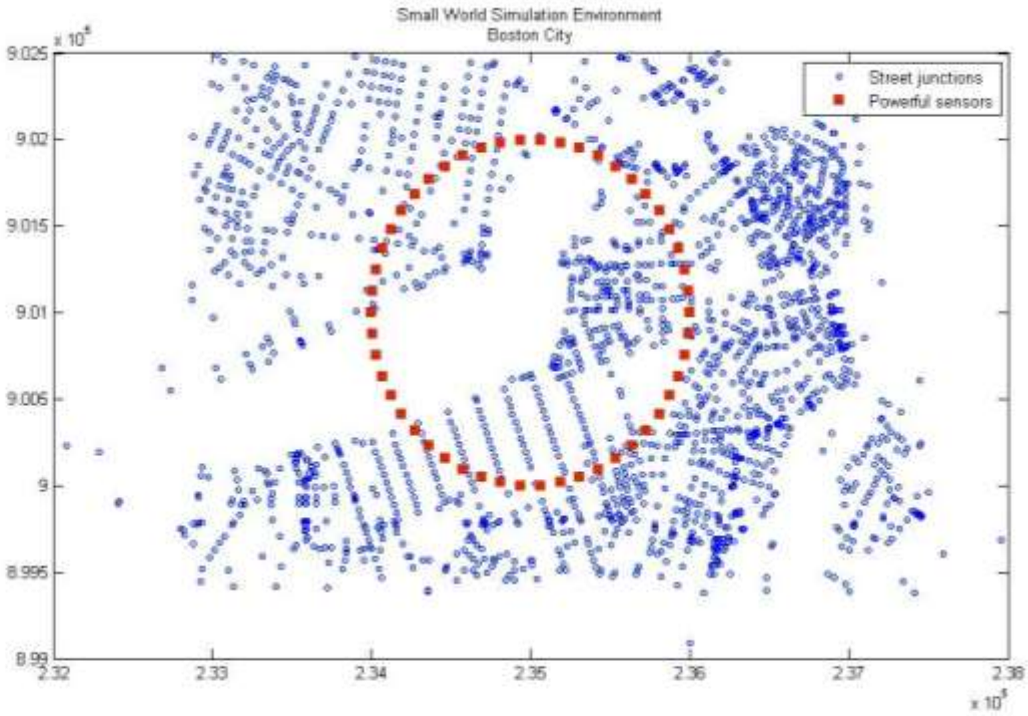


Fig. 6.3 Powerful sensor placement diagram

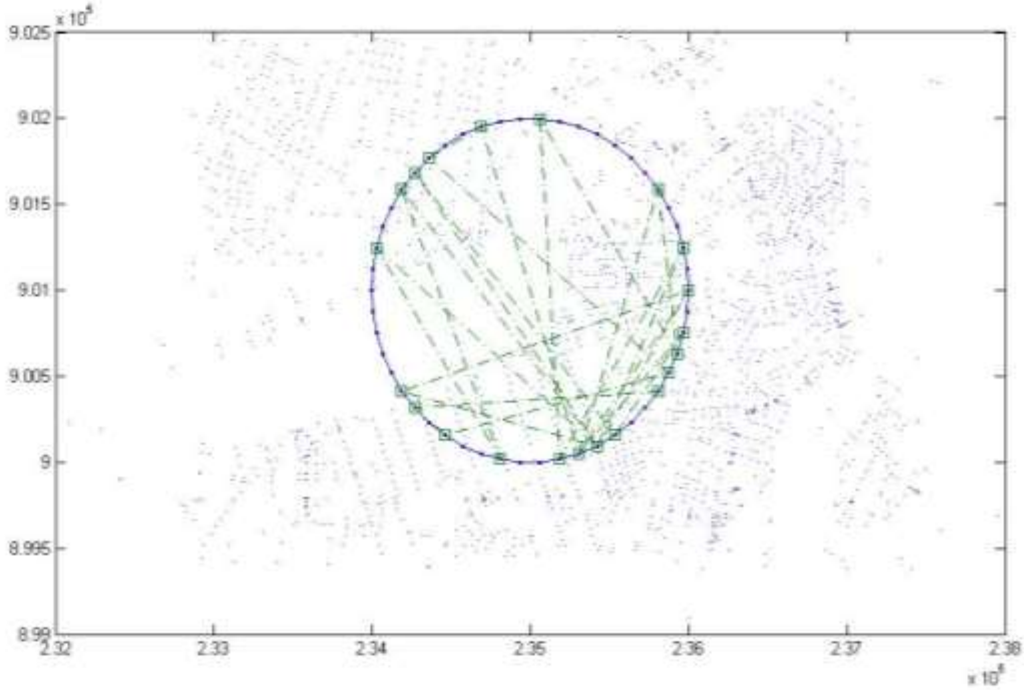


Fig. 6.4 Small-world connection (re-wiring probability β= 0.01)

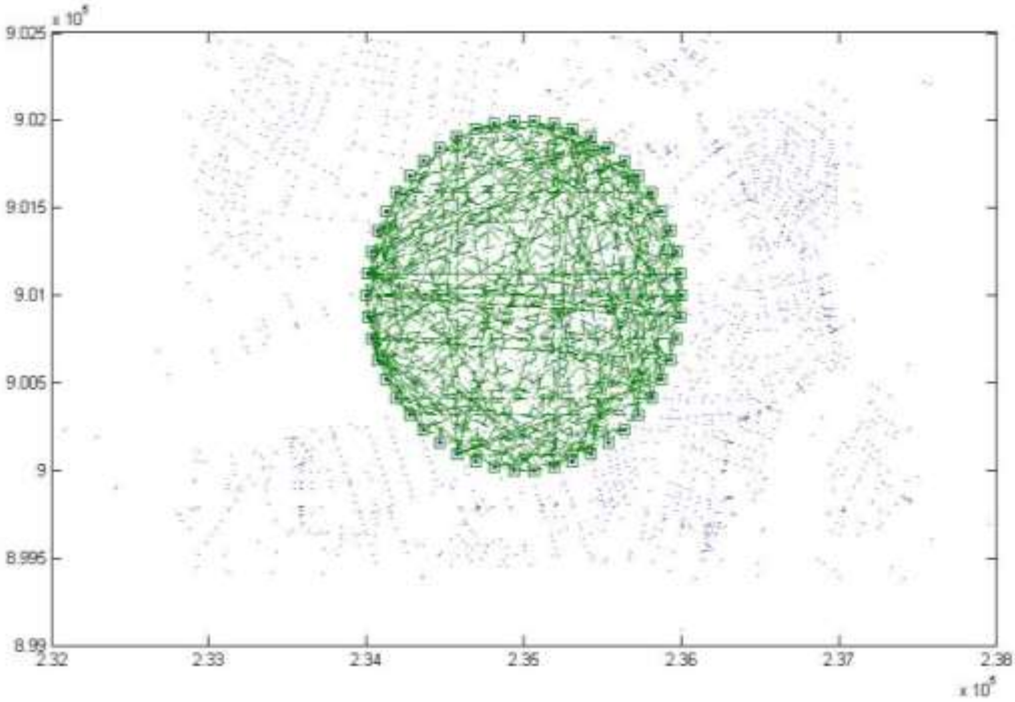


Fig. 6.5 Small-world connection (re-wiring probability β = 0.1)

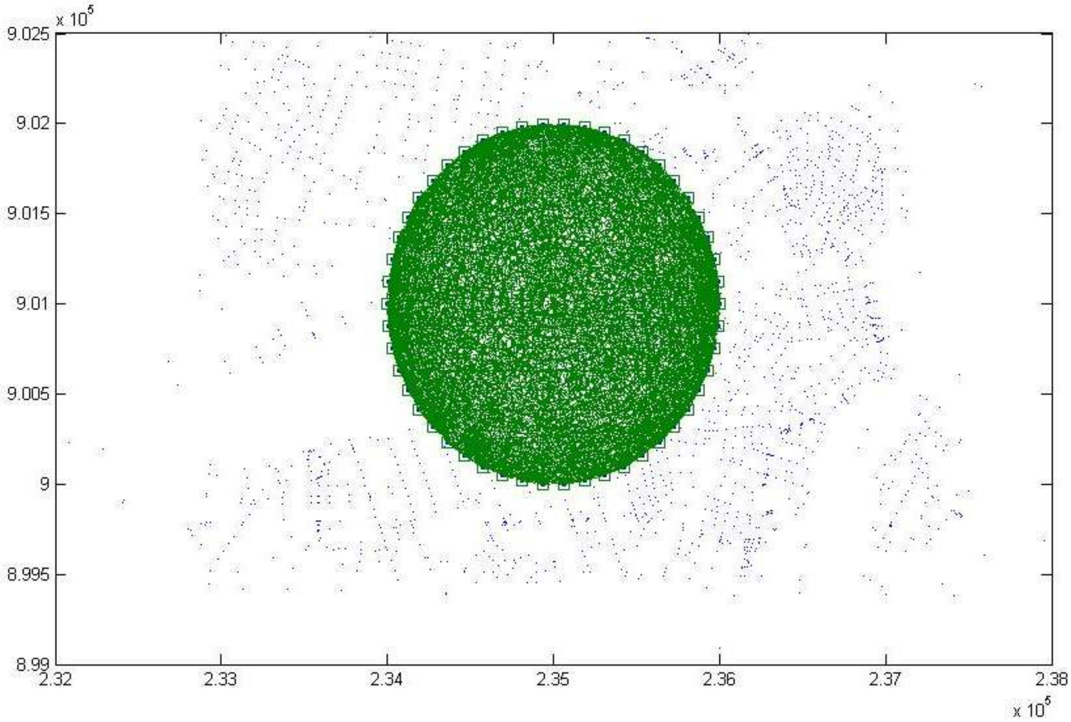


Fig. 6.6 Small-world connection (re-wiring probability β=0.6)

The simulation results are shown in table 6.1:

Re-wiring probability β	0.01	0.1	0.6
Average distance (in kilometer)	3.75	3.62	3.50
Average node hop[1]	9.42	7.33	5.26
Average Energy cost (in mW)[2]	9420	16417	22361
Small-World Network Energy cost (in W)[3]	520	5200	31200
Bit Error Rate (in bps/mW)[4]	9.42×10^{-4}	8.54×10^{-4}	5.21×10^{-4}

Table 6.1 Simulation results of three small-world network scenarios

The average node hop contains both the wireless sensor node hops and the small-world sensor node hops, so it is calculated as Eq. 6.4:

$$n_{hop} = \frac{\sum_{i=1}^{N}(n_{i,WSN}+n_{i,SM})}{N} \tag{6.4}$$

In Eq. 6.4, n_{hop} represents the average node hop, $n_{i,\,WSN}$ represents the number of i^{th} connection pair randomly selected from 2031 WSN nodes, and $n_{i,\,SM}$ represents the number of Small-World powerful sensor nodes connecting the i^{th} connection pair.

The average energy cost is derived from average node hop values, by multiplying WSN node hops with WSN antenna transmitter power [6.7] and multiplying small-world node hops with wireless base station antenna transmitter power [6.8] using equation 6.5:

$$E_{hop} = \frac{\sum_{i=1}^{N}(n_{i,WSN} \times P_{WSN} + n_{i,SM} \times P_{SM})}{N} \quad (6.5)$$

In Eq. 6.5, P_{WSN} represents the antenna transmitting power of WSN nodes, and P_{SM} represents the antenna transmitting power of Small World network nodes. The average node energy cost E_{hop} consists of these two power values. In this simulation, P_{WSN} is set to 100 mW and P_{SM} is set to 20 W. These values are based on wireless transmission standard IEEE 802.11b [6.9], where the maximum transmission power of a wireless antenna is set to 100 mW, and the maximum transmission power of a base station antenna is set to 20 W.

The small-world network energy cost is computed from base station antenna transmitter power [6.8]. It only contains the transmission energy consumed by small-world base stations.

Based on the results shown in table 6.1, a conclusion can be made that β = 0.01 is better compared with 0.1 and 0.6, the reasons are listed below:

When the value of β increased, the average distance of random communications within the network did not alter a lot. The small-world network topology did not affect the geographic distance. However, the average hop count between two randomly selected nodes dramatically decreased. This is because powerful sensor nodes from the small-world network attended this communication session. The quality of communication can be improved because less number of hops was used in communication. So conclusion one: increasing the value of re-wiring parameter β could improve the signal quality of communications.

A disadvantage of increasing the value of β is the energy consumption of the total network. As is shown in row 4 and row 5 of table 6.1, the average energy cost of random communications within the network, as well as the total network power consumption is almost linearly proportional to the value of β. The transmitting power of one small-world network station is 20 times of one wireless sensor antenna. Thus, an energy-optimal deployment strategy shall not include many network stations.

In order to evaluate the efficiency of implementing small-world network on top of an existing Boston city sensor network, an indicator of signal quality improvement per energy unit is calculated in row 6 of Table 6.1. The value of this indicator is calculated as the signal quality (BER) of each connection divided by the total energy cost, as is shown in equation 6.6:

$$BER_{SM} = \frac{\sum_{i=1}^{N}(n_{i,WSN} \times BER_{WSN} + n_{i,SM} \times BER_{SM})}{N} \tag{6.6}$$

The value of BER_{WSN} is 10^{-4}/100m, which is set as same as previous simulations. The value of BER_{SM} is set according to IEEE 802.11b standard [6.8], which is 10^{-6}/100m.

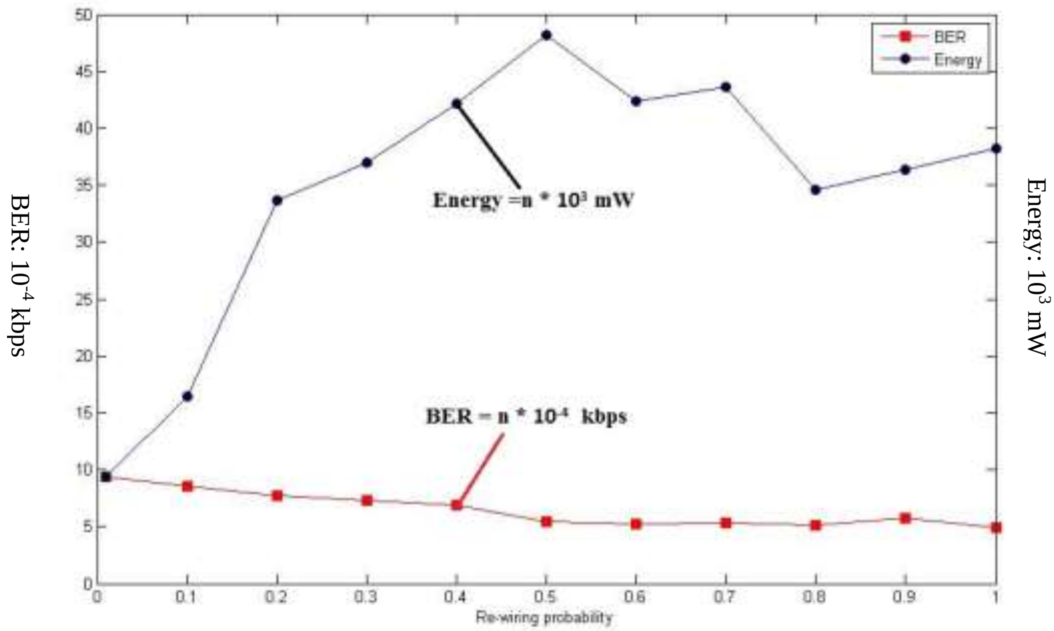


Fig. 6.7 Curve of BER (in red) and Energy (in blue) versus Re-wiring probability β

From Figure 6.6 one can observe that with increase of the re-wiring probability β, the average signal quality is improving because its bit error rate is decreasing. However, the energy cost is increasing. The energy cost curve fluctuates within range β= [0.4, 1.0], indicating that the average energy spent on random connections comes to its maximum value. Further increase of the re-wiring probability value does not deteriorate energy consumption in this scenario. Thus, the research focus shall be placed in range β = [0.01, 0.4].

In figure 6.7, the relationship between the signal quality efficiency and the re-wiring probability β is shown in the curve. As is clearly observed in this graph, the signal quality efficiency quickly shrinks within range [0.01, 0.4], and fluctuates within range [0.4, 1]. An optimal value of β shall satisfy the requirement that its energy efficiency is the maximum of the curve. Thus, β = 0.01 is selected. (While β = 0.001 is better than β = 0.1, the quantity of re-wiring connections [$0.001 \times 50 \times 51 = 2.55$] is too few for a simulation, so β = 0.01 is selected).

In table 6.2, three values are used to describe the difference between three simulation models. The details are explained in the following:

The average hop count in EP and SG simulations are set manually: in these simulations, the maximum distance between two transmitting nodes is set as 720 meters, with 72 meters a hop, so the average hop count of EP and SG is 10.

In SG simulation, only data loss rate is evaluated, so 0.3771 is the average data loss rate of a connection whose path length is 720 meter. In order to convert this rate into bit error rate, the physical layer information of the communication channel needs to be provided.

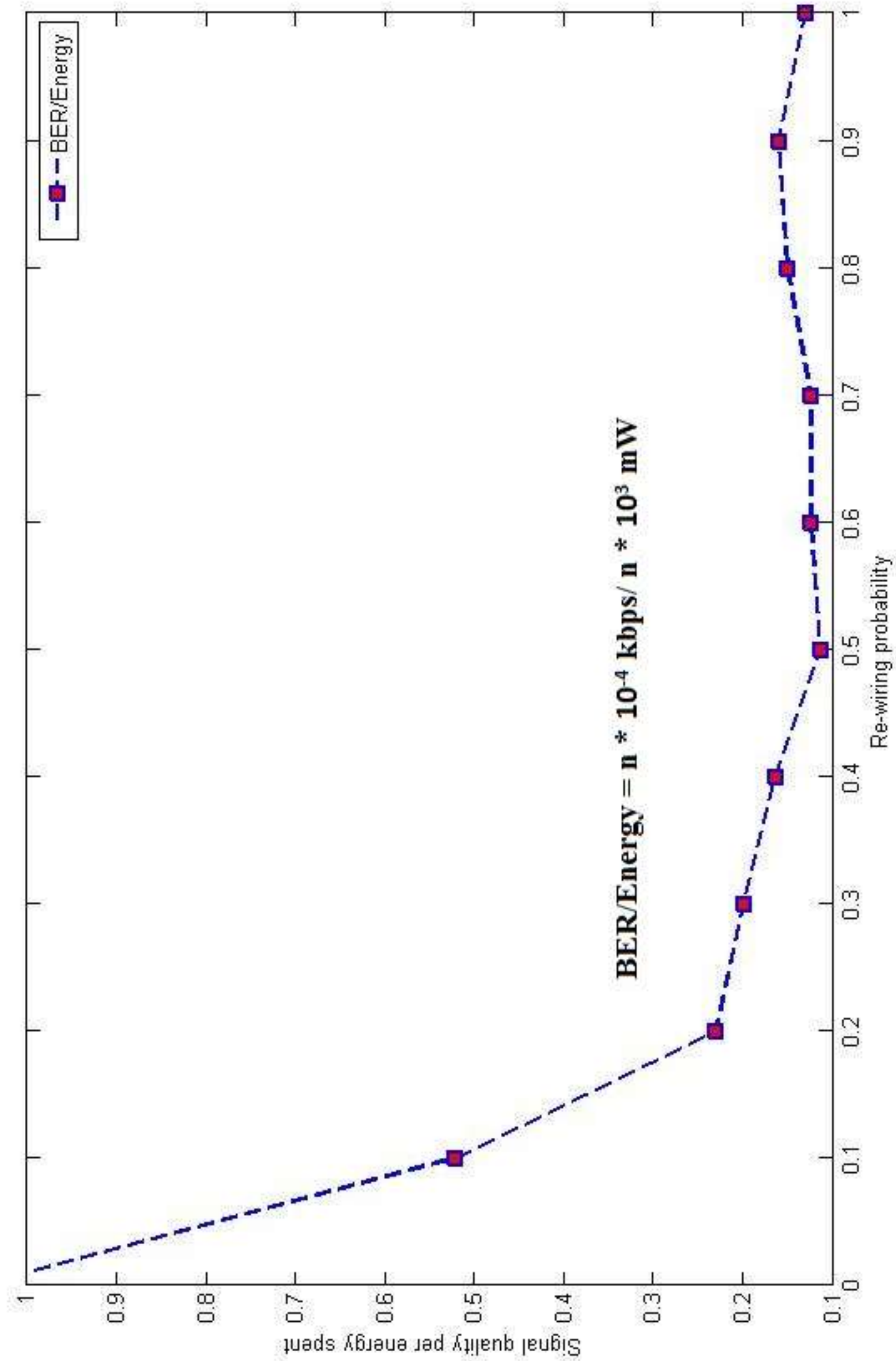


Fig. 6.8 Signal quality efficiency versus re-wiring probability β

Simulation Model	**SW**	**EP**	**SG**
Average Hop Count	7.33	10	10
Average Energy Consumption (mW)	16417	8133	7453
Average Bit Error Rate (bps)	8.54×10^{-4}	3.3×10^{-2}	0.38

Table 6.2 Comparison results of EP, SG, and Small-World simulations

In conclusion, the simulation results from table 6.2 show the advantage of the small-world network is to improve the signal quality in exchange of additional power stations. As is shown small world network model utilizes fewer hop counts to transmit signals covering the whole Boston city. Because the number of intermediate hop counts decreased, the transmission latency is greatly improved. In addition, interferences caused by routing devices are decreased, which bring improvement of signal quality. However, the transmission power of the newly added powerful sensors and base stations brings high energy consumption of the network. Thus, the small world network is suitable to work on high-performance networks. However when the network power is limited, WSNs performs better.

6.4 CONCLUSIONS

In this chapter, an introduction to the small-world network model is presented, followed by its simulation based on the Boston city map. A series of simulations applying different values of re-wiring probability β are executed, with detailed results and discussions followed. Considering the power consumption required by the small-world network, an energy-efficient implementation with $\beta = 0.1$ is selected as a benchmark model. Comparison results of this benchmark simulation with EP and SG models are presented in Table 6.2. From this result, it can be concluded that the small-world topology ($\beta = 0.1$) is an efficient approach to improve the average end to end signal quality transmitted throughout an urban wireless network. However, taking into account of its power consumption, it is less efficient than previous EP model because it introduces additional network sensors which consume 20 times more energy than WSN sensors. The advantage of small-world network topology is that it simplifies the urban network by reducing the average hop number greatly. From the simulation results it is possible to observe that the average number of hops can be reduced into 6, which means any end to end transmission request of Boston city can be connected with 6 intermediate sensors. This will greatly benefit the real-time applications, which request not only quality of service but also less time latency. From this point of view, small-world network is suitable for implementation within modern urban cities.

REFERENCES

6.1. S. Milgram, The small world problem, Psychology Today 1, 61, 1967.

6.2. D. J. Watts and S. H. Strogatz, Collective dynamics of 'small-world' networks. Nature 393: 440-442. 4th June 1998.

6.3. P. Erdos and A. Renyi, On the evolution of random graphs, Publication of the Mathematical Institute of the Hungarian Academy of Sciences, pp. 17-61. 1960.

6.4. F. Zaidi, Small world networks and clustered small world networks with random connectivity, Social Network Analysis and Mining, vol. 3, N. 1, p. 51-63, Mar. 2013.

6.5. Z. W. Xu, and S. Daniel, Small-world characteristics on transportation networks: a perspective from network autocorrelation, Journal of Geographical Systems, vol. 9, no. 2, pp. 189-205. Apr. 2007.

6.6. L. C. Freeman, Centrality in social networks: conceptual clarification, Social Networks, 1(3), pp. 215-239. 1979.

6.7. M. Krunz, A. Muqattash, and S. J. Lee, Transmission power control in wireless ad hoc networks: challenges, solutions and open issues. Network, IEEE, Vol. 18, Issue 5, Sept.2004.

6.8. Mahasukhon, P., et.al, IEEE 802.11b based ad hoc networking and its performance in mobile channels, Communications, IET, vol. 3, issue 5, May 2009.

6.9. D. G. Yoon, S. Y. Shin, et.al, Packet Error Rate Analysis of IEEE 802.11b under IEEE 802.15.4 Interference, Vehicular Technology Conference, pp. 1186—1190. 2006.

CHAPTER 7

VANET ADVANCEMENTS AND FUTURE WORK

In this chapter an introduction of future advancements of VANET and potential applications of optimization algorithms is presented. It consists of four sections: section 7.1 introduces VANET evolution which allows mobile nodes and wireless sensors multicast capability and channel transmission diversity. In section 7.2, a report is given on recent progresses of swarm intelligence (SI), including particle swarm optimization (PSO) and ant colony optimization (ACO). Two industrial applications—public transportation congestion control and mobile cloud computing, are analyzed in section 7.3 with EP and SW tools. From these practical scenarios one can observe how to implement optimization algorithms on problems in our daily life. To conclude, optimization in future VANET environment requires a multicast version, and multiple objectives EP, PSO and ACO can be implemented to analyze and optimize it. In addition, in many real-world applications, such as transportation planning and mobile cloud computing, optimization algorithms are still effective. Thus, optimization in VANETs has a splendid future in tomorrow's research.

7.1 HARDWARE ENHANCEMENTS

In this section a brief introduction about VANET hardware improvements is presented, with a particular focuses on the following equipment upgrades:

- Orthogonal Frequency Division Multiplexing (OFDM) [7.1]

- WAVE (Wireless Access in Vehicular Environment) [7.2]
- Receiver diversity [7.3]

The OFDM system (shown in Fig. 7.1) provides a wireless LAN with a data transmission capability of 5, 10, 20 and 30 Mbps. It uses 48 subcarriers modulated with Binary or Quadrature Phase Shift Keying (BPSK/QPSK), or 16-Quadrature Amplitude Modulation (16-QAM). Forward error correction coding is used with a coding rate of ½ or ¾.

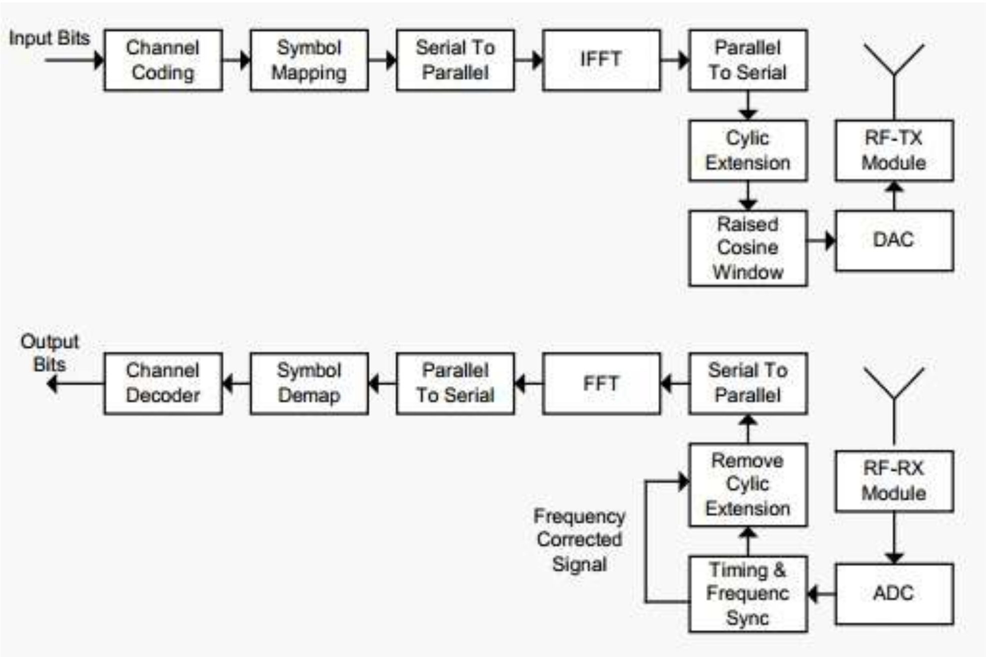


Fig. 7.1 OFDM block diagram [7.1]

Research evaluating OFDM performance in a standard VANET environment [7.4] reports that a receiver can configure 1024 adjacent channels (901 of them used in communication) with 5 MHz bandwidth operating in the 2.4 GHz frequency band [7.5]. The probability of correct detection of incoming channels reaches 90% when an average SNR of transmitting signals is above

20 dB (as shown in Fig. 7.2). In addition, the overall delay period reaches 200 μs for each symbol transmitted in the system.

Implementation of the OFDM system on the Boston city scenario will alter the network model because now the OFDM-enhanced transmitters can configure and connect with multiple routing devices (sensors). With all VANET nodes upgraded to an OFDM mode, they can configure multiple paths during movement and connect with their neighbours through multiple active routes. The network architecture would be similar that of the small world model [6.3] mentioned in chapter 6. To analyze this model and optimize the routes' combination, a multicast routing strategy is needed.

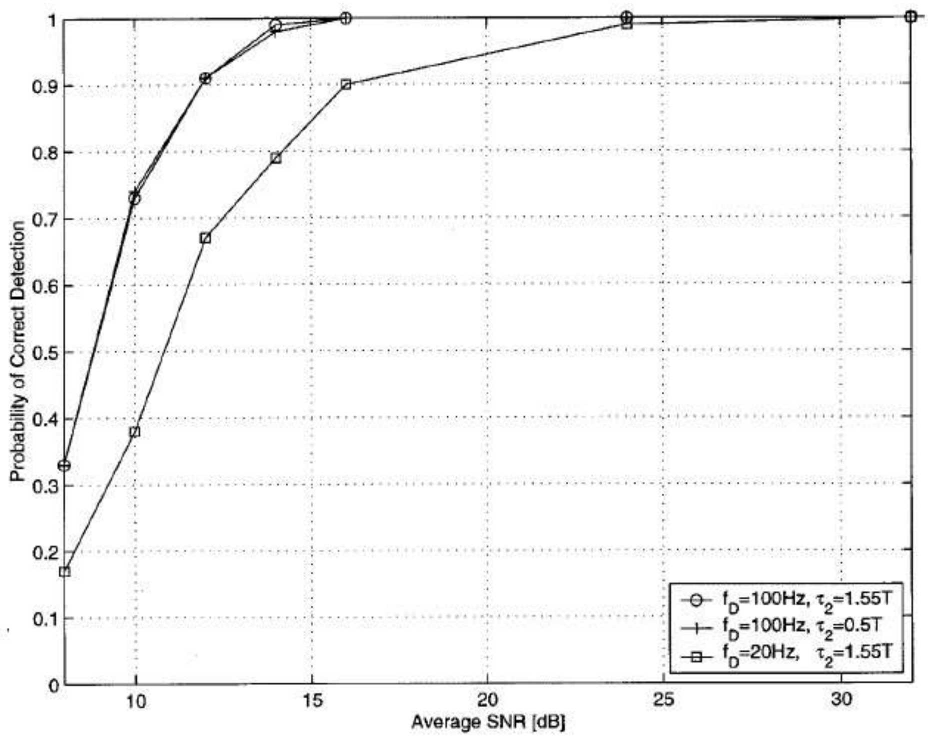


Fig. 7.2 Probability of correct detection of the paths in OFDM scenario [7.5]

In July 2010, an international standard (IEEE 802.11p) was designed especially for a VANET, and given the name **WAVE**: Wireless Access in Vehicular Environments. In its physical layer specification [7.2], the IEEE clearly stated that all WAVE receivers should implement OFDM: orthogonal frequency division multiplexing technology. In fig. 7.3, a block diagram of a dual antenna WAVE receiver is shown.

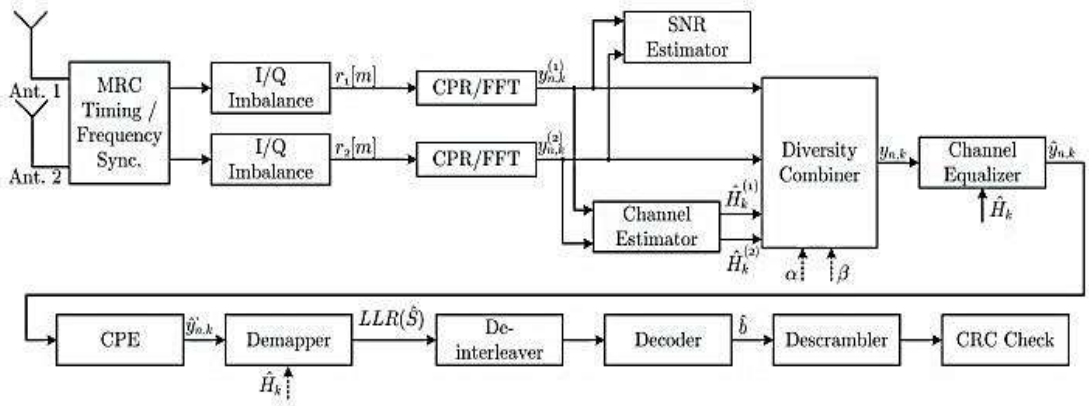


Fig. 7.3 Dual antenna WAVE block diagram [7.6]

Based on the simulation results in report [7.6], the implementation of the dual antenna architecture into VANETs reduces the frame error rate (FER) by 25% or gives a diversity gain of 2 dB when the FER is 10%, compared with the single antenna case. In the same comparison, the dual antenna WAVE system requires an additional synchronization procedure, and a channel diversity combiner, with a better performance in SNR and energy conversation from the transmitter side.

In order to fit in this dual antenna multicast VANET architecture, the optimization algorithms need to adapt new features. SG is advantageous in this scenario because it depends on the Poisson distribution density λ, which is not related to the architecture of the network itself. In practice, a dual antenna hardware layer at transmitter side can effectively reduce transmit power and improve the signal quality, so less hops are required by the SG to cover the same distance for the previous unicast version. In addition, the OFDM technique allows multiple channels to access one mobile node at the same time. This allows SG to further reduce the data loss rate because interference level between multiple channels is decreased. Therefore, SG is an optimization technique for WAVE.

7.2 SWARM INTELLIGENCE TECHNIQUES

In section 7.2, the following optimization techniques will be introduced:

- Particle Swarm Optimization [PSO] [7.10]
- Ant Colony Optimization [ACO] [7.11]

Swarm intelligence (SI) was firstly introduced in research [7.8], and its applications on wireless sensor networks can be found in a survey [7.9]. It is a summary of various optimization schemes such as Particle Swarm Optimization (PSO) [7.10], Ant Colony Optimization (ACO) [7.11], Artificial Bee Colony (ABC) [7.12], and so on. Algorithms with SI are inspired by the behaviors of real population (ants, bees, etc.) and try to evolve with rules observed from these colonies. The following paragraphs discuss these SI techniques and their possible implementation on the Boston scenario.

PSO is a stochastic optimization technique mimicking the swarm behavior of birds flocking. In execution, each potential solution is named as a particle, with its fitness value defined as a particle position and a velocity, depending on its gap from global optima. The steps needed in PSO are shown in below:

- Particle and memory initialization
- Velocity updating
- Particle position updating
- Memory updating
- Repeat updating process until termination criteria satisfied

Compared with EP, PSO introduces another parameter called particle velocity to identify the speed of an optimization to its threshold value. By evaluating the updated velocity parameter of the new particles, PSO can recognize if this optimization iteration is successful or not. This evaluation process can either be knowledge-based, or a self-training progress. In the Boston simulation, PSO can be implemented the same way as multiple objectives EP was. The velocity parameter can be defined as a hop count between two communicating nodes or as an angle from current hop to its destination. In this case, an optimization process becomes a clustering approach such that all vehicles are heading to the destination node.

The ACO is inspired by the food-seeking behavior of real ants. In simulation, a population is randomly scattered across a map, and a transit probability $p_k(r,s)$ is defined in eq. 7.1 from node r to node s for the k^{th} ant:

$$p_k(r,s) = \begin{cases} \frac{[\tau(r,s)][\eta(r,s)]^{\beta}}{\sum_{\mu \in J_k(r)}[\tau(r,u)][\eta(r,u)]^{\beta}} & if\ s \in J_k(r) \\ 0, & otherwise \end{cases} \quad \text{Eq. 7.1}$$

In Eq. 7.1, $\tau(r,s)$ is the pheromone level between node r and node s, $\eta(r,s)$ is the inverse of the distance between node r and node s, $J_k(r)$ is the set of nodes that remain to be visited by the k^{th} ant positioned on node r, and β is a parameter determined by the relative importance of pheromone level versus distance. The algorithm repeats with the probability $P_k(r,s)$ until the ant has visited all the nodes of the map. This procedure is called initialization. An update procedure is defined in Eq. 7.2:

$$\tau(r,s) = (1-\alpha)\cdot\tau(r,s) + \sum_{k=1}^{m} \Delta\tau_k(r,s),$$

$$\Delta\tau_k(r,s) = \begin{cases} \frac{1}{L_k} & if\ (r,s)\epsilon\ route\ visited\ by\ ant\ k \\ 0 & otherwise \end{cases}, \qquad \text{Eq. 7.2}$$

In Eq. 7.2, $\Delta\tau_k(r,s)$ is the pheromone level laid down between node r and s by the k^{th} ant, L_k is the length of the route visited by the k^{th} ant, m is the number of ants and α is a pheromone decay parameter. In order to locate an optimal route, ants are moved to a designated city and repeat Eq. 7.1 and Eq. 7.2, until the behavior clearly unveils a condition that all ants are traveling through this route.

In the Boston city simulation, an ACO algorithm could be used to find an optimal repeater placement architecture. A group of ants is randomly placed across the street map, and parameter τ is assigned as the data loss rate, and η as the inverse value of the distance between two neighbouring nodes. Let the ants visit all the nodes within Boston city repeatedly with procedures defined by Eq. 7.1 and Eq. 7.2 repeatedly. The number of visited ants on each street or street junction is recorded in order to discover the most popular street set (or junction set), and repeaters are placed only on this optimal set.

In summary, section 7.2 introduces SI which contains many interesting optimization techniques inspired from biology and nature. Two algorithms named PSO and ACO, are analyzed, which reference the flocking behavior of birds and an ant colony food seeking mode. PSO is similar to the multiple objectives EP algorithm, because it has two parameters to evolve: particle position and particle velocity. However, these parameters are individually evolved from one generation to another, while in multiple objectives EP they are expressed in a uniform fitness function, and evolve according to a single fitness value. ACO is similar to a multicast routing algorithm, because multiple ants are seeking their shortest paths to their destinations simultaneously. Its evolution is decided by the amount of information that the ant colony can collect during path seeking. From this point of view, ACO is suitable for optimal sensor placement generation. ACO can be implemented on a Boston map to locate information-maximum street combination, and place sensors along these streets to reduce energy cost and maximum transfer efficiency.

7.3 APPLICABILITY OF EP, SG AND SW

This section addresses the applicability of three optimization techniques: EP, SG, and SW. These applications focus on the following research fields:

- Public transportation congestion control
- Mobile cloud computing

In public transportation system design, an outstanding issue is traffic congestion and how to arrange people to travel within a public transportation system (e.g. subways, coach) to avoid unnecessary trip delay or congestion at certain area is a hot research topic. In simulation, an optimization algorithm can collect information from sensing units placed around stations and analyze the traffic load of different timeslots. With these data available, the algorithm can give prediction of the future traffic load and make adjustments on transportation timetable.

In an existing research report [7.13], a GA was used to search for a near-optimal traffic signal timetable based on samples collected from a microscope traffic simulator. Three different traffic loads: low, medium and high volumes were simulated. The results showed that GAs can optimize traffic loads in low and high volume scenarios. The EP operator is similar to a GA, so will probably produce similar output. In conclusion, a well-designed EP could optimize public traffic congestion situation.

Mobile cloud computing [7.14] is defined as an infrastructure where both the data storage and data processing happen within a *cloud environment* outside of the mobile device. The cloud environment includes Internet, private and public platforms, and a wireless network consisting of other mobile devices. In Fig. 7.4, mobile devices are connected to the mobile networks through wireless connections to base stations. Mobile network service providers deliver mobile-specific services from mobile networks, including connecting the mobiles to a cloud. In the cloud, services are provided via Internet gateways.

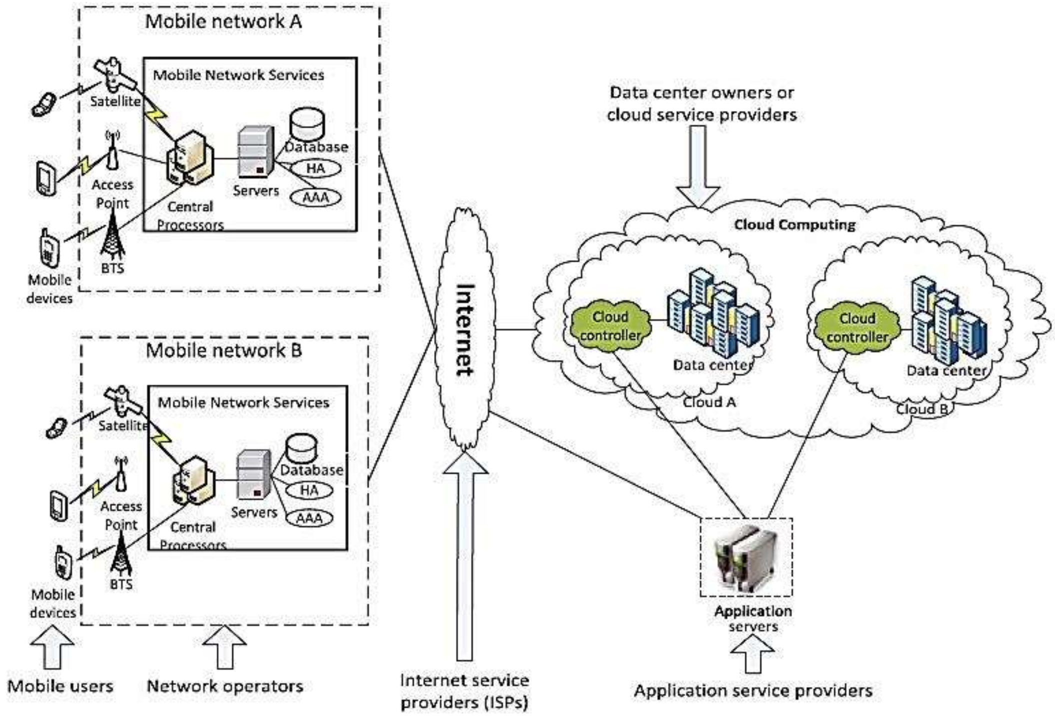


Fig. 7.4 Mobile cloud computing architecture [7.13]

In a technical survey [7.14], a research focus on mobile cloud computing is the limitation of its bandwidth. Current mobile network standards support a maximum bandwidth of 14.4 Mbps. However, cloud computing requires sufficient amount of mobiles to attend in sessions simultaneously, and this will bring network congestion. SW can provide a potential solution on this issue. In simulation, powerful cloud computing servers can be placed around service data centres, and they can connect multiple mobiles and monitor congestion status. By choosing a proper re-wiring parameter β, and optimizing the number of these powerful cloud computing servers, a mobile cloud computing platform can be generated.

In this section, two potential application fields were put forward for optimization algorithm's implementation: public transportation congestion control and mobile cloud computing. In the first case, a research based on genetic algorithm was cited to evidence that EP is capable to analyze and optimize traffic congestion problems. In the second scenario, the author proposed an approach based on SW to solve low bandwidth problem, which was identified by researchers as a major technical issue in mobile computing. Based on these investigations, a conclusion is made that optimization techniques mentioned in this thesis are applicable to in future research frontiers.

7.4CONCLUSIONS

In this chapter, three aspects to clarify the future of optimization algorithms in VANETs have been considered. First of all an introduction of next generation's VANET was presented in section 7.1. The network architecture named WAVE has been standardized as IEEE 802.16p. Within WAVE, two essential techniques are embedded: OFDM and receiver diversity. In summary, tomorrow's VANET should have multiple antennas and multicast capability. In section 7.2, SI was introduced as future of optimization algorithms. Two of its variants: PSO and ACO were analyzed, together with author's comments on their possible implementations on a Boston city simulation. Finally, in section 7.3, two important research issues were raised: public transportation congestion control and mobile cloud computing bandwidth efficiency. It is recommended that EP and SW to be applied in these scenarios to provide optimization solutions.

To conclude, an optimization algorithm for the next generation of VANET should support multiple antenna transceiver configurations and support multicast scenarios. In this thesis, EP is a unicast version, but SG and SW can be modified easily into multicast versions. In addition, SI can provide alternative optimization approaches to investigate VANET problems. Two scenarios have been put forward to show how to implement PSO and ACO in a Boston city simulation. Finally, two real-world applications are analyzed with EP and SW to show the applicability of the optimization techniques. From the discussions and facts stated above, the author can announce that EP, SG and SW optimization techniques are highly adaptable for future VANET implementation, and they can be applied widely in many research areas.

REFERENCES

7.1. Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specification Amendment 6: Wireless Access in Vehicular Environments. IEEE 802.11p standard, IEEE, July 15, 2010.

7.2. Richard van Nee, OFDM Physical Layer Specification for the 5 GHz Band, Lucent Technologies Bell Labs Research, IEEE 802.11-98/12, January 1998.

7.3. Christian Hoymann, Analysis and performance evaluation of the OFDM-based metropolitan area network IEEE 802.16. vol. 49, issue 3, pp. 341-363, Computer Networks, October 2005.

7.4. Overall Requirements on the Radio Interface(s) of the UMTS, ETSI SMG, ETSI, Tech. Rep. ETR/SMG-21.02, v3.0.0, Valbonne, France, 1997.

7.5. Yang B. G., Khaled B. L., Roger S. C. and Cao Z. G., Channel estimation for OFDM transmission in multipath fading channels based on parametric channel modeling, vol. 49, no. 3, IEEE transactions on communications, Mar. 2001.

7.6. Shivaldova, V., et.al, Performance Evaluation of IEEE 802.11p infrastructure-to-vehicle real-world measurement with receive diversity. Wireless Communications and Mobile Computing onference, pp. 848-852. July 2011.

7.7. El-Keyi, A. et.al, MIMO VANETs: research challenges and opportunities, Computing, Networking and Communications (ICNC), pp. 670-676, Jan. 2012.

7.8. Engelbrecht Andries P, Fundamentals of computational swarm intelligence, Wiley, 2005.

7.9. Kulkarni, R. V., Computational Intelligence in Wireless Sensor Networks: a survey. Vol. 13, issue 1, communications surveys & tutorials, IEEE, 2011.

7.10. Kennedy, J., Eberhart, R., Particle swarm optimization. Proc. IEEE Conf. on Neural Networks, pp. 1942-1948, IEEE Press, New York, 1995.

7.11. Dorigo, J. M., et.al, Ant Colony System: optimization by a colony of cooperating agents. IEEE Trans. On Systems, Mans and Cybernetics—Part B: Cybernetics 26, pp. 29-41, 1996.

7.12. Karaboga, D, An idea based on honey bee swarm for numerical optimization, technical report-TR06, Erciyes University, Computer Engineering Department, 2005.

7.13. Byungkyu Park, et.al, Traffic signal optimization program for oversaturated conditions: a genetic approach. Journal of transportation research record, vol. 1683, pp. 133-142, Jan. 2007.

7.14. H. T. Dinh, C. Lee, et.al, A survey of mobile cloud computing: architecture, applications, and approaches. Wireless Communications and Mobile Computing, vol. 13, issue 18, pp. 1587-1611, Dec. 2013.

7.15. W. G. Song, X. Zhou, et.al, Review of Mobile cloud computing, Communication Software and Networks (ICCSN), 2011 IEEE 3rd International Conference, pp. 1-4, May 2011.

Appendix A: Publication List

Xiang Yu, Dr. Mark Leeson and Prof. E. L. Hines, Urban Ad Hoc Network Performance Optimization, Journal of AISS (Advances in Information Science and Service Sciences), vol.5, no.11, June 2013.

Xiang Yu, Dr. Mark Leeson and Prof. E. L. Hines, A Novel Stochastic Model for VANET Optimization, peer reviewed and now is in submission stage, Oct. 2013.

Xiang Yu, Dr. Mark Leeson and Prof. E. L. Hines, Dynamic traffic routing optimization in DWDM using genetic algorithms, Optical Networks, 2010 (peer reviewed), Oct 2010.

Appendix B: Boston city data set

The Boston city data set is extracted from an image file ‘Boston.tif’. This image is a visible red, green and blue composite from a georeferenced IKONOS-2 panchromatic/multispectral product created by GeoEyeTM. This Boston map is part of MATLAB resources and can be found in local MATLAB folder (*matlabroot\toolbox\map\mapdata*). For more information one can refer to the MATLAB Tutorial: *Tour Boston with Map Viewer APP*.

Below is a list of Boston street junctions’ geographical coordinates (in meter), in total 2031 nodes. The data is stored in ‘NodeXY.xlsx’. The street length information is contained in ‘Node_Edges.xlsx’. The distance matrix is contained in ‘AdjacentMatrix.xlsx’.

Node ID	X	Y		Node ID	X	Y
0	232085.4063	900229.8751		1015	235754.2813	900159.5001
1	232286.1875	900193.5001		1016	235757.5157	899641.4375
2	232408.3437	899893.1875		1017	235758.0937	900237.5001
3	232411.2187	899910.6251		1018	235758.3437	901075.6875
4	232683.2657	900683.1251		1019	235763.6251	900307.6875
5	232738.6875	900553.9375		1020	235767.6407	900064.0625
6	232798.6563	899748.7501		1021	235768.3595	899688.0625
7	232813.4687	899741.4375		1022	235771.3751	900813.5001
8	232824.9845	899712.4375		1023	235774.2813	902393.7501
9	232869.3751	901076.0001		1024	235774.3751	902398.0001
10	232872.8595	901160.0001		1025	235774.5469	902404.9375
11	232877.4845	899777.6875		1026	235775.7031	902263.6875
12	232880.1563	899662.0625		1027	235776.9531	901859.4375
13	232880.8281	902017.6875		1028	235778.4375	899743.9375
14	232889.3125	901802.1875		1029	235779.8125	899759.6251
15	232900.9219	899774.1251		1030	235781.7813	899778.6251
16	232908.5313	899931.2501		1031	235785.2969	902252.9375
17	232911.5313	899790.5625		1032	235787.9375	901153.7501
18	232911.5469	899950.5001		1033	235788.4063	901840.1251
19	232914.2657	901851.6251		1034	235788.4845	901867.5625
20	232921.7813	901651.8751		1035	235790.7031	902271.5001

21	232929.3125	900057.2501		1036	235793.0313	899605.1251
22	232929.4531	899446.6251		1037	235794.9375	899881.8125
23	232933.0469	899518.8125		1038	235795.6875	900233.3125
24	232933.2031	899787.8751		1039	235799.6719	901849.8125
25	232935.5157	900109.1875		1040	235800.7031	902259.5625
26	232950.2657	901720.3125		1041	235805.7345	901831.5625
27	232953.3907	901486.1251		1042	235818.3437	899475.0001
28	232956.0937	901701.8125		1043	235820.0469	901824.1251
29	232973.1407	901766.1875		1044	235821.0157	901280.3125
30	232975.0625	900052.2501		1045	235821.1095	902212.3751
31	232975.9845	901974.3751		1046	235822.5625	901413.9375
32	232979.6095	901362.3125		1047	235822.5781	899617.5001
33	232983.4687	899585.3125		1048	235822.6563	901445.0625
34	232983.6407	901464.1875		1049	235824.1875	901465.3751
35	232990.0781	902004.6251		1050	235825.1719	901817.8125
36	232993.3437	901594.5001		1051	235827.7187	900885.5001
37	232997.7187	902319.5625		1052	235827.8281	900060.7501
38	233006.7969	900973.1875		1053	235828.2501	900929.3125
39	233010.1563	902042.8751		1054	235828.3751	901154.5625
40	233010.6251	901238.3125		1055	235829.1719	900967.1251
41	233011.6563	901580.1875		1056	235829.9219	901019.8751
42	233012.5937	899626.7501		1057	235831.2969	902298.1251
43	233023.1875	901672.0001		1058	235831.3437	901076.6875
44	233023.5469	900681.8125		1059	235832.5157	902214.5001
45	233028.4845	901427.6251		1060	235834.9845	899670.6875
46	233029.4219	902487.2501		1061	235837.3595	900835.3751
47	233029.5001	902079.5625		1062	235842.9531	899981.0625
48	233041.0469	902297.9375		1063	235846.5937	900155.1875
49	233044.7031	899668.6875		1064	235848.4063	902282.3751
50	233045.0157	901264.8125		1065	235854.2657	899561.4375
51	233047.5937	901733.8125		1066	235859.6251	900932.8751
52	233049.6563	901943.6251		1067	235862.3125	900226.0625
53	233051.5937	900615.8751		1068	235862.7969	901783.0625
54	233064.8437	901975.6875		1069	235865.6719	902320.9375
55	233065.0001	902449.1875		1070	235866.6407	900895.7501
56	233070.5469	901790.1251		1071	235868.2657	902322.1251
57	233072.3125	901289.3751		1072	235875.3595	900301.1251
58	233075.8125	899708.6251		1073	235876.0625	899659.9375
59	233077.9219	901808.0625		1074	235879.3125	901283.8751
60	233080.9375	902009.8751		1075	235882.0781	901332.5625
61	233082.0157	901576.6251		1076	235886.1095	902304.1251
62	233085.8437	901504.5001		1077	235886.1251	899711.1875
63	233089.3437	901721.6251		1078	235887.2501	901078.1875
64	233093.4845	901845.8751		1079	235888.8907	899724.0625

65	233096.9531	902044.4375		1080	235888.9063	902305.6875
66	233101.3751	901220.4375		1081	235890.6095	899741.3751
67	233101.4687	899511.2501		1082	235895.0625	902226.9375
68	233102.7345	901652.0625		1083	235897.3437	899721.6251
69	233105.7657	901773.1251		1084	235899.7969	899794.6251
70	233108.0157	901534.3125		1085	235901.7031	900413.5625
71	233117.0157	899899.1251		1086	235902.0625	901737.5625
72	233117.8281	901353.4375		1087	235904.3595	901851.4375
73	233118.7969	901790.3125		1088	235904.3751	900220.8125
74	233120.5001	901712.3751		1089	235905.6875	901848.3125
75	233120.6095	899919.3751		1090	235906.4219	901521.1251
76	233121.7969	899773.3751		1091	235906.4531	901284.3751
77	233124.3907	901426.3751		1092	235908.7813	901836.6251
78	233125.0469	901914.6875		1093	235910.6563	900255.3751
79	233134.9531	899418.5001		1094	235911.5625	899859.3751
80	233134.9531	899788.8751		1095	235913.0001	901837.6251
81	233139.5157	900089.1875		1096	235914.9845	901812.9375
82	233141.5313	901335.4375		1097	235915.5781	901809.5001
83	233148.9063	902169.7501		1098	235916.5625	900296.0625
84	233149.4845	902357.9375		1099	235923.9375	902273.0001
85	233152.6251	901497.3751		1100	235924.5625	901820.6251
86	233153.0469	900036.3751		1101	235925.1407	901081.2501
87	233155.0781	901531.8125		1102	235925.5313	899424.5001
88	233157.8751	901571.0625		1103	235926.6407	901817.5001
89	233166.1719	902008.0001		1104	235926.9375	899934.2501
90	233166.6095	899977.6875		1105	235927.4845	902274.4375
91	233167.0157	901895.8125		1106	235927.7657	901391.6251
92	233167.0781	899829.3125		1107	235934.4531	900294.0001
93	233171.2031	901631.3125		1108	235936.0313	900251.8125
94	233173.6251	901164.4375		1109	235936.7969	901285.3751
95	233176.4845	899889.9375		1110	235938.3751	900405.1875
96	233186.2657	901694.6875		1111	235938.6875	901512.0001
97	233190.7813	899910.0625		1112	235941.5625	900344.3751
98	233191.5313	902450.3751		1113	235941.8595	902258.2501
99	233199.4063	899886.3751		1114	235944.3907	902259.9375
100	233200.7969	901288.6875		1115	235951.5313	901150.4375
101	233202.5937	901877.7501		1116	235955.2031	901084.3125
102	233202.7657	901623.1251		1117	235956.1251	899530.5625
103	233203.5937	902297.5625		1118	235956.5469	902345.6251
104	233206.2345	901350.1251		1119	235957.1095	899852.6251
105	233207.7031	902113.6251		1120	235959.0469	901588.1251
106	233209.9845	900759.4375		1121	235959.6719	902347.5001
107	233212.5001	901418.0001		1122	235959.8437	901841.8751
108	233218.5625	901490.0001		1123	235962.1407	900294.1875

109	233218.8281	902227.1251		1124	235964.9375	902347.3125
110	233225.6563	901566.3751		1125	235965.3437	901613.3125
111	233232.2187	902171.0625		1126	235966.9375	900479.8125
112	233232.6719	901859.3751		1127	235966.9531	900337.3751
113	233232.9219	901263.6251		1128	235969.5001	899770.2501
114	233236.8281	900699.8125		1129	235970.6251	901855.6251
115	233240.3751	901356.4375		1130	235972.9219	899920.7501
116	233241.8125	901966.7501		1131	235975.1719	901789.1875
117	233244.9531	901413.4375		1132	235977.0313	900615.5001
118	233246.0781	899968.1875		1133	235977.9531	899809.5625
119	233251.4845	900671.9375		1134	235981.0781	901284.8751
120	233255.9845	901676.4375		1135	235982.0001	900294.1875
121	233259.6407	902237.5001		1136	235984.2501	899848.1251
122	233261.1563	902429.0001		1137	235985.7501	899630.2501
123	233270.4531	902059.3125		1138	235987.0469	900897.6251
124	233270.7969	901729.6875		1139	235988.3595	899874.6251
125	233281.3437	901227.3125		1140	235992.3595	900401.6251
126	233283.5625	899920.8125		1141	235994.2345	902407.5001
127	233287.6719	900098.5625		1142	235994.3125	899913.0625
128	233295.4219	901822.0001		1143	235996.7345	900433.6875
129	233303.0469	902257.5625		1144	235998.1875	902209.9375
130	233313.4687	902302.6875		1145	235999.2187	901664.6251
131	233313.6719	901248.1251		1146	236000.9219	899675.8125
132	233313.9063	899590.8125		1147	236004.1407	899686.8751
133	233316.6563	900806.0625		1148	236005.1095	902211.7501
134	233317.4063	902174.2501		1149	236005.1875	899093.4375
135	233322.4687	901348.0625		1150	236005.7657	900865.6251
136	233325.3437	901924.4375		1151	236006.3437	901284.3751
137	233328.3751	901405.8125		1152	236006.4375	899696.1251
138	233331.5157	900085.8125		1153	236006.7031	902401.4375
139	233335.1563	901480.5001		1154	236007.5937	900138.8125
140	233343.3595	902355.4375		1155	236009.6563	899990.2501
141	233343.5469	901992.3751		1156	236010.2969	901090.3751
142	233343.8907	900069.8125		1157	236010.3125	899710.3125
143	233344.0001	902241.1251		1158	236011.5157	901059.4375
144	233344.8125	900854.2501		1159	236012.3907	900667.3751
145	233345.5157	900018.2501		1160	236016.0469	899729.6875
146	233352.1407	900002.4375		1161	236019.9063	899669.6251
147	233354.5001	900719.9375		1162	236024.2345	900204.6875
148	233357.4531	900827.8751		1163	236027.8595	902391.3751
149	233360.9063	901904.8125		1164	236030.3751	900923.2501
150	233363.1719	902284.1875		1165	236031.1095	899899.7501
151	233365.2813	901279.1251		1166	236031.4531	900247.1875
152	233368.6251	899925.8751		1167	236032.5781	902419.1875

153	233369.2345	899890.2501		1168	236032.7657	902209.8125
154	233370.1251	902383.3125		1169	236033.2345	902182.1251
155	233371.2031	901781.8751		1170	236033.4531	902169.5625
156	233372.6251	899953.6875		1171	236034.4219	900044.6251
157	233372.9063	899804.6875		1172	236034.9531	900577.6251
158	233375.6563	901153.3125		1173	236035.0313	899797.5001
159	233376.6875	902108.5625		1174	236039.0781	900303.6875
160	233384.0469	902338.6251		1175	236041.7657	900381.5001
161	233384.0937	901474.8125		1176	236043.4375	900401.5625
162	233386.4219	901948.7501		1177	236045.4687	899689.3751
163	233386.8595	900769.4375		1178	236046.7969	899836.3751
164	233387.7501	902424.1875		1179	236048.1095	901700.1875
165	233393.9531	900017.5001		1180	236048.2345	900889.3125
166	233395.9375	902372.0625		1181	236052.0781	902353.3125
167	233399.5001	901837.6251		1182	236053.8125	902376.5001
168	233401.0937	899418.6875		1183	236056.2031	900562.0625
169	233403.3751	902207.5001		1184	236058.0157	900512.9375
170	233409.5001	902072.5001		1185	236058.7813	901393.0625
171	233418.1563	901874.3125		1186	236061.3281	899670.1251
172	233419.2657	901981.1251		1187	236063.0469	899888.2501
173	233420.2031	899708.0625		1188	236063.0625	900733.5625
174	233424.3125	900100.6251		1189	236065.5781	902369.0625
175	233428.6251	899533.0625		1190	236066.4845	900462.0001
176	233432.5313	901747.3125		1191	236068.2501	900128.0001
177	233436.3437	901322.8125		1192	236073.6251	900471.6251
178	233439.5625	901904.0625		1193	236075.9531	901324.3751
179	233443.6251	902353.1875		1194	236076.5781	902361.6251
180	233458.2657	902018.8751		1195	236077.2031	900201.7501
181	233463.2969	901805.1251		1196	236078.8281	902384.5001
182	233463.8125	901215.9375		1197	236080.1251	899946.5625
183	233467.5781	901935.1875		1198	236081.3281	900146.4375
184	233467.6095	902342.4375		1199	236084.7657	900123.5625
185	233470.5157	901535.6875		1200	236084.7969	901548.5625
186	233472.8125	901466.6875		1201	236086.0001	899378.6875
187	233480.1407	901841.5001		1202	236087.6875	900763.2501
188	233488.6563	901011.1251		1203	236088.0625	902380.8125
189	233490.5469	902246.1251		1204	236089.5157	899767.8125
190	233490.5469	902332.1875		1205	236093.7657	900309.1875
191	233490.9063	901513.6875		1206	236094.7657	900500.8751
192	233497.3751	900176.9375		1207	236095.1251	902313.9375
193	233498.8907	901709.0625		1208	236096.4687	901733.8125
194	233500.3751	902209.6875		1209	236097.9845	900007.8125
195	233500.5937	901540.3125		1210	236098.8281	900777.5001
196	233502.2969	901967.5625		1211	236099.2031	900264.6875

197	233508.2187	900962.5625		1212	236100.4845	900658.5625
198	233511.2187	899637.1875		1213	236104.8751	899817.9375
199	233512.0625	901565.1875		1214	236105.3281	901351.0625
200	233515.9687	902144.6251		1215	236106.9375	902216.6875
201	233517.4063	900133.5001		1216	236107.9531	901425.2501
202	233517.9219	900938.1251		1217	236110.2813	900522.3751
203	233524.7187	902100.3751		1218	236110.4687	902029.6875
204	233525.0313	900180.2501		1219	236110.6407	901998.0001
205	233526.3751	900161.0625		1220	236113.8437	899846.1875
206	233527.5001	900136.5001		1221	236119.5001	902029.6875
207	233527.8125	899655.1875		1222	236120.1251	902333.3125
208	233528.0157	902317.4375		1223	236121.5001	900700.6875
209	233528.3437	900097.1875		1224	236121.5157	900643.4375
210	233530.0001	901815.8751		1225	236122.6563	901068.3125
211	233532.2187	900184.4375		1226	236123.0781	901010.2501
212	233534.9375	902058.8751		1227	236123.4219	900963.6251
213	233536.7657	901622.0625		1228	236124.6563	901262.0625
214	233537.9219	902149.1251		1229	236125.1719	899493.4375
215	233538.3437	900072.6875		1230	236125.5157	899727.6875
216	233542.3281	900022.7501		1231	236127.5937	900110.9375
217	233543.4375	900178.6875		1232	236129.0001	900900.3751
218	233544.2969	899931.0001		1233	236133.3751	900131.7501
219	233544.7345	899895.8751		1234	236134.0313	902342.5001
220	233550.8751	901804.6251		1235	236134.0781	899537.1251
221	233551.2657	901991.6251		1236	236134.6563	902460.3125
222	233552.6563	900844.3751		1237	236135.2187	900384.4375
223	233554.9375	900184.1875		1238	236135.2187	901378.3125
224	233556.3907	901672.3751		1239	236136.1875	901957.7501
225	233556.7969	901207.5625		1240	236138.5469	899538.0001
226	233558.2187	900831.4375		1241	236141.0937	900413.8125
227	233559.6407	902221.8751		1242	236141.5157	900314.4375
228	233560.3907	900825.5001		1243	236141.5157	901766.3125
229	233564.4375	900813.3125		1244	236143.4063	901532.6251
230	233569.1563	900030.8751		1245	236144.0781	900957.8125
231	233572.7657	899793.1251		1246	236147.5001	900839.5625
232	233575.4845	901167.0001		1247	236147.6407	899635.6875
233	233576.6095	901888.4375		1248	236147.9375	900426.6875
234	233576.8751	902434.5001		1249	236148.1875	900743.3751
235	233577.2031	900034.5001		1250	236148.3437	901936.0001
236	233577.8751	900003.1875		1251	236148.7657	902442.0001
237	233579.1875	899896.7501		1252	236149.0157	899487.3125
238	233579.1875	899931.8751		1253	236149.2969	899587.9375
239	233580.4531	899793.1251		1254	236149.9219	900573.5625
240	233580.6563	901955.2501		1255	236150.6563	900259.0625

241	233580.8907	899776.0001		1256	236150.9531	899652.7501
242	233581.5781	899747.1251		1257	236151.0157	899980.3751
243	233581.9063	899737.0625		1258	236151.7813	901840.2501
244	233585.6563	899687.3125		1259	236152.2969	899540.5625
245	233585.9063	900037.2501		1260	236152.9375	899585.8125
246	233586.7813	899994.8125		1261	236153.8595	900682.1875
247	233587.7187	899932.3125		1262	236153.9219	900202.8751
248	233588.5625	899897.1875		1263	236154.8281	899678.1251
249	233590.2813	899793.1251		1264	236155.7187	901244.5625
250	233590.9063	900151.0001		1265	236155.8437	902355.6251
251	233591.3437	899747.1251		1266	236157.2657	899690.6251
252	233591.5625	899775.1251		1267	236158.3437	899696.1251
253	233593.5469	900039.6251		1268	236159.8595	900856.0625
254	233595.3437	899704.1875		1269	236161.5781	899615.5625
255	233595.3595	902037.3751		1270	236162.5781	901556.9375
256	233595.5937	899793.1251		1271	236162.7813	901954.2501
257	233599.8595	901779.1251		1272	236163.4219	899828.3125
258	233600.3437	900152.7501		1273	236164.0001	899631.6251
259	233602.0469	900069.0625		1274	236164.5001	900317.3751
260	233602.6719	900041.9375		1275	236164.5937	899579.0625
261	233611.2501	902075.3751		1276	236164.7969	899542.0625
262	233619.0625	900117.4375		1277	236165.4063	899631.2501
263	233622.9531	901450.8751		1278	236166.8437	899650.3751
264	233623.6251	901837.4375		1279	236167.0157	899542.2501
265	233623.8281	900110.4375		1280	236168.8751	900502.8125
266	233624.8751	902107.9375		1281	236170.0313	899650.1875
267	233632.7501	899651.1251		1282	236171.2969	899561.9375
268	233640.9687	900075.0001		1283	236171.5313	899676.5625
269	233643.1251	900055.6875		1284	236172.9063	901452.0001
270	233643.3595	902154.3751		1285	236172.9219	899570.4375
271	233646.9687	899991.4375		1286	236173.5313	900712.1875
272	233647.9375	899935.7501		1287	236173.8281	902493.4375
273	233650.1251	900123.6251		1288	236174.6095	899692.8751
274	233650.5001	899898.5001		1289	236175.4375	901292.7501
275	233654.6719	899748.0625		1290	236176.3125	899675.8751
276	233655.2813	901614.0001		1291	236179.2187	899805.0625
277	233655.3281	902180.8751		1292	236179.6875	899693.6875
278	233659.6095	900225.9375		1293	236183.1875	899783.3125
279	233666.4845	901210.0625		1294	236184.4063	900087.0001
280	233668.7501	901943.0001		1295	236184.4219	900752.1251
281	233668.8125	901741.8125		1296	236184.7345	901797.0625
282	233676.6251	899635.2501		1297	236185.7969	899627.5001
283	233683.6563	902250.5625		1298	236186.4375	899821.6875
284	233689.7501	901444.1251		1299	236188.4531	900290.0625

285	233691.1719	901997.5001		1300	236189.4687	900266.0001
286	233698.5313	899993.6251		1301	236191.2657	899647.8125
287	233706.3751	902031.5625		1302	236192.9845	899550.3125
288	233708.3437	899641.7501		1303	236193.1407	899550.1251
289	233709.6407	900162.8751		1304	236194.6875	901516.1875
290	233713.3751	899794.8751		1305	236194.8907	901862.8125
291	233715.6251	902321.1251		1306	236196.2657	900321.0625
292	233719.1251	899413.3751		1307	236197.4063	902288.1875
293	233719.6251	902066.8125		1308	236197.5469	899910.8751
294	233727.7031	900924.8751		1309	236198.5001	899674.6875
295	233727.7657	899556.6251		1310	236200.2345	900225.4375
296	233738.2969	901710.6875		1311	236200.5625	899954.2501
297	233742.5937	900893.5625		1312	236200.5625	900392.0001
298	233742.8751	902113.8125		1313	236201.7187	901867.2501
299	233749.7657	899645.5625		1314	236204.0001	899696.0625
300	233756.7657	902145.0001		1315	236205.8281	901593.4375
301	233760.8437	899999.3125		1316	236208.8281	900191.4375
302	233767.1563	899941.6875		1317	236209.5001	901815.3125
303	233771.2969	899903.7501		1318	236209.8437	900648.9375
304	233772.3281	901435.6251		1319	236216.4375	900798.4375
305	233773.4845	900244.1875		1320	236216.6563	899624.6875
306	233781.0469	901688.0625		1321	236218.0781	901877.6875
307	233784.0625	902213.5001		1322	236218.7969	900936.9375
308	233785.5937	900216.0001		1323	236218.8437	899761.2501
309	233791.3595	900201.0625		1324	236219.1875	901841.3125
310	233794.8751	900191.8751		1325	236221.7345	901001.5625
311	233800.2969	900177.5625		1326	236222.4687	901824.1251
312	233812.9063	901636.3125		1327	236223.0781	899643.6251
313	233815.2969	902285.8125		1328	236224.9063	901811.2501
314	233819.7345	901669.0001		1329	236225.2657	900820.0001
315	233820.4845	899516.3751		1330	236225.7345	901882.5625
316	233821.7969	900120.3751		1331	236226.6251	900186.2501
317	233831.9845	901732.3751		1332	236227.7345	902447.6251
318	233832.2969	899573.4375		1333	236229.2501	901828.3751
319	233838.2187	900077.2501		1334	236230.0937	900433.0625
320	233840.0781	899616.0001		1335	236231.4063	901886.3751
321	233843.1251	899680.5001		1336	236232.1407	900789.0625
322	233843.2969	899633.3125		1337	236233.8751	900067.3125
323	233853.4063	900032.4375		1338	236234.4219	899674.5625
324	233853.9845	902382.3125		1339	236235.5313	899677.6251
325	233868.5313	899993.0625		1340	236238.9845	899699.6875
326	233875.7031	902202.7501		1341	236238.9845	899756.4375
327	233878.4063	901570.2501		1342	236241.3125	900322.3751
328	233886.0313	899947.5001		1343	236243.8907	901838.1251

329	233890.6095	901833.0625		1344	236246.0937	900809.0001
330	233892.4219	899930.1251		1345	236247.2969	901840.1875
331	233893.3281	902483.5001		1346	236247.9531	899702.5625
332	233894.7031	899924.0625		1347	236251.3437	900219.9375
333	233895.7031	900262.6251		1348	236252.9375	901635.5625
334	233899.4845	899911.3125		1349	236252.9531	899493.0625
335	233905.0469	901007.1875		1350	236254.3595	901843.5625
336	233916.9375	901863.5625		1351	236259.0157	899992.6875
337	233917.9063	899856.9375		1352	236260.2187	901847.3751
338	233919.9845	901871.6875		1353	236260.5157	899998.5001
339	233921.8125	901427.1251		1354	236260.9063	900831.1251
340	233926.0937	899626.7501		1355	236262.1407	900174.8125
341	233930.9063	899818.6251		1356	236262.2657	901091.1251
342	233931.9687	902319.0625		1357	236262.9375	899827.4375
343	233933.6407	901936.2501		1358	236265.2813	900321.4375
344	233946.3751	900246.9375		1359	236266.3281	900055.7501
345	233948.6563	902006.9375		1360	236272.2031	900591.7501
346	233949.3907	899566.5001		1361	236272.2657	899996.1251
347	233950.0781	899769.9375		1362	236272.7187	900053.8751
348	233953.4531	899637.6875		1363	236273.6719	900000.8751
349	233955.7345	899755.5625		1364	236277.5469	901490.2501
350	233959.4845	902081.5001		1365	236279.4063	900037.1875
351	233967.0157	899726.9375		1366	236279.9845	900911.8751
352	233968.7501	901134.0001		1367	236281.0313	899675.6875
353	233969.6875	902391.6251		1368	236281.4687	900051.3125
354	233979.0625	901839.1875		1369	236284.2969	900200.2501
355	233979.9845	901583.8751		1370	236284.3751	900050.4375
356	233981.2345	899685.0625		1371	236287.6407	900059.9375
357	233984.7813	902232.1251		1372	236287.6563	899847.9375
358	233992.2969	899653.2501		1373	236287.7657	900134.5001
359	233993.1875	901576.7501		1374	236287.9687	899842.5001
360	233996.7031	902309.4375		1375	236288.2657	899834.6875
361	233999.4687	899636.1251		1376	236288.6407	900167.1251
362	234006.0781	899986.3751		1377	236289.7969	899843.4375
363	234006.6407	900254.5001		1378	236289.9375	900088.8125
364	234009.8907	901053.8125		1379	236290.3281	900048.6875
365	234010.6875	901364.3125		1380	236290.5157	901867.2501
366	234011.6563	902384.8751		1381	236292.6719	899836.0625
367	234015.6251	901418.0625		1382	236294.6251	901773.2501
368	234023.7187	899573.8751		1383	236295.0937	901674.3125
369	234025.1251	900196.1875		1384	236295.4531	900165.3751
370	234025.2031	901021.9375		1385	236296.6875	900749.3125
371	234026.6407	899946.7501		1386	236298.1563	901756.5625
372	234026.8437	902461.5625		1387	236301.2969	900163.6875

373	234036.3595	901362.5001		1388	236302.4063	900407.2501
374	234040.6719	899755.5001		1389	236304.4845	900334.1875
375	234042.2969	900151.2501		1390	236304.8437	900162.6875
376	234044.3595	899518.9375		1391	236304.8751	900317.0625
377	234049.2657	902222.9375		1392	236305.6563	899850.5001
378	234051.2501	899759.3751		1393	236305.8281	900901.2501
379	234051.8595	899802.2501		1394	236306.8437	899837.3751
380	234052.2345	899755.7501		1395	236308.7813	900161.6251
381	234054.1251	899625.8751		1396	236312.5469	901684.8751
382	234057.7969	902151.3751		1397	236313.7187	900641.9375
383	234058.8437	900107.9375		1398	236315.6563	899855.1875
384	234059.2657	899729.9375		1399	236317.3907	899836.1251
385	234062.3751	902300.3751		1400	236317.7657	900774.6875
386	234064.0001	899466.5625		1401	236319.7657	900158.3125
387	234067.7969	899690.5625		1402	236321.5001	900836.5001
388	234068.2031	901183.1875		1403	236324.6719	901545.3751
389	234069.9219	901701.5625		1404	236324.7657	900611.3751
390	234074.0157	900066.2501		1405	236324.9063	901693.8125
391	234075.1875	902373.0625		1406	236325.0937	901952.7501
392	234081.1095	899769.9375		1407	236327.0001	900156.0625
393	234083.4063	901339.5625		1408	236327.8281	899749.7501
394	234087.9845	902451.0625		1409	236328.9375	899742.0001
395	234089.4531	901410.8751		1410	236330.2657	900952.6875
396	234090.4531	900017.5001		1411	236332.2345	900538.0625
397	234096.4531	901821.3751		1412	236336.3595	901701.2501
398	234099.2813	902216.6875		1413	236338.3125	900676.3125
399	234107.3281	901905.1875		1414	236338.9375	900034.3751
400	234107.4687	899975.1251		1415	236340.0625	899750.8125
401	234111.2969	902290.7501		1416	236340.8281	900458.4375
402	234123.3907	902365.4375		1417	236342.1563	901050.7501
403	234123.4063	901980.7501		1418	236344.9219	901708.3751
404	234124.1875	899933.1875		1419	236346.3595	901518.1251
405	234128.8125	902021.0625		1420	236350.2031	901959.9375
406	234131.0781	900300.3751		1421	236350.4845	900880.5001
407	234133.8751	902055.1875		1422	236352.6095	900815.0625
408	234137.2031	902442.9375		1423	236361.7813	900943.3125
409	234139.9375	899890.7501		1424	236363.3437	900310.5625
410	234144.2969	899792.6875		1425	236363.5937	901525.2501
411	234147.9531	902129.6875		1426	236364.3595	901921.8751
412	234150.5157	899772.0625		1427	236364.9375	901043.6875
413	234151.0937	900241.3125		1428	236367.0469	900834.3125
414	234152.6407	901221.2501		1429	236367.7345	900581.6251
415	234154.0001	899846.0625		1430	236369.2031	900287.8751
416	234154.9845	899758.0001		1431	236369.7345	900280.8125

417	234157.3437	899730.9375		1432	236369.9687	902014.6251
418	234158.9375	902205.6875		1433	236372.1875	900498.5001
419	234167.1563	900196.0001		1434	236372.3751	901079.9375
420	234168.0157	901402.8125		1435	236372.9375	900405.3125
421	234168.1095	899801.2501		1436	236373.5937	900140.9375
422	234170.1095	901479.0625		1437	236375.0001	901766.7501
423	234171.9845	902279.9375		1438	236375.7969	900715.6875
424	234174.6719	901808.2501		1439	236377.2031	901734.6875
425	234181.8437	900155.0625		1440	236378.7813	901110.3125
426	234184.4687	902354.2501		1441	236379.2969	900434.0625
427	234188.0781	902011.3751		1442	236380.2657	900255.1875
428	234192.0781	899665.6875		1443	236381.0937	901428.6251
429	234198.1407	900112.0001		1444	236381.7813	900602.2501
430	234198.3751	902431.9375		1445	236383.7501	901656.5625
431	234201.6407	901140.3125		1446	236385.3595	901637.0001
432	234201.9687	901626.6251		1447	236387.2657	901629.4375
433	234215.5781	900066.0001		1448	236387.9375	900294.5001
434	234217.6407	901728.0625		1449	236389.2187	900862.0625
435	234225.0625	899593.5001		1450	236392.4375	900018.3125
436	234227.6095	901800.8125		1451	236393.5625	901600.0625
437	234232.8125	900020.7501		1452	236393.7031	901637.6875
438	234236.2345	901440.3751		1453	236395.0313	901584.6251
439	234237.4375	901260.5001		1454	236396.2501	901402.2501
440	234240.8751	901603.3125		1455	236396.5157	900754.9375
441	234242.6251	901882.0001		1456	236397.9375	900302.0001
442	234243.8437	899588.3125		1457	236399.6563	901077.6875
443	234247.5001	899980.0001		1458	236400.1719	900232.0625
444	234248.0625	901383.5001		1459	236400.4375	900279.4375
445	234257.8907	901957.1251		1460	236401.1251	900879.3751
446	234260.6095	899528.3751		1461	236403.9531	901958.8125
447	234264.1251	899936.6875		1462	236404.2031	900300.3125
448	234266.4531	901412.9375		1463	236405.2657	901612.0625
449	234270.4375	902030.2501		1464	236406.5313	901536.5001
450	234276.6251	901175.9375		1465	236407.2657	900071.5001
451	234278.9687	899890.3125		1466	236408.8595	901912.6875
452	234280.6719	901513.5625		1467	236408.9375	901445.7501
453	234283.0781	902105.8125		1468	236410.0937	901816.1251
454	234289.8125	900360.0625		1469	236411.3751	901598.8125
455	234291.5469	901142.1251		1470	236412.3125	900896.6251
456	234292.8281	899847.0625		1471	236412.4063	900287.4375
457	234295.5625	902181.2501		1472	236412.4063	900648.9375
458	234308.4375	902257.6251		1473	236412.6407	900297.4375
459	234310.9219	900301.4375		1474	236414.0001	900793.2501
460	234311.1719	901567.1251		1475	236414.1719	900296.8751

461	234320.7657	902332.1251		1476	236414.4687	900298.5001
462	234321.9531	901158.8125		1477	236415.5157	900304.3125
463	234325.7501	899572.5001		1478	236415.9219	900128.5001
464	234326.7031	901357.4375		1479	236415.9531	900454.8751
465	234327.0937	900255.5625		1480	236416.7345	900311.0001
466	234335.4687	902410.0001		1481	236416.9531	901563.0625
467	234342.4375	900212.0625		1482	236417.4845	901562.5625
468	234350.2657	899394.5625		1483	236418.2813	900323.0625
469	234353.5157	901212.6251		1484	236421.2345	901782.9375
470	234354.5469	899391.1875		1485	236422.3281	899921.0001
471	234359.7187	900168.2501		1486	236423.3907	900293.6251
472	234361.1251	899567.3751		1487	236424.4375	901927.5001
473	234369.2501	901706.8125		1488	236425.6875	901842.5625
474	234376.2969	900124.8751		1489	236426.5469	901740.4375
475	234392.8125	900079.6875		1490	236427.3907	901641.1251
476	234393.5937	901358.7501		1491	236428.0157	901543.5001
477	234398.2345	902136.1875		1492	236428.2501	901030.4375
478	234398.9219	901778.3125		1493	236429.0625	901468.0001
479	234400.4375	901234.3125		1494	236429.5937	900410.1251
480	234406.1875	900038.6875		1495	236433.0313	901577.2501
481	234407.0469	902009.6875		1496	236433.5313	901543.5625
482	234419.7813	902084.1875		1497	236434.3907	900223.0625
483	234421.6251	901374.7501		1498	236435.2969	900589.6875
484	234421.8907	899870.8125		1499	236436.3595	900434.2501
485	234422.2345	899990.5625		1500	236436.3907	900062.4375
486	234427.9687	902133.8125		1501	236437.3125	900792.0625
487	234429.1875	901850.5625		1502	236438.1719	900810.4375
488	234432.0313	902158.4375		1503	236440.2657	901317.8751
489	234436.4845	899950.3125		1504	236441.2501	901818.8751
490	234437.2031	901322.1875		1505	236445.5469	901855.9375
491	234444.8281	902235.2501		1506	236445.7031	900003.0625
492	234452.1251	899906.1875		1507	236450.9063	901279.0625
493	234456.7501	901237.3125		1508	236452.2813	900583.1875
494	234456.7657	902310.1251		1509	236452.8281	901542.5625
495	234459.3907	901924.9375		1510	236456.8751	901537.3751
496	234460.9687	899516.8125		1511	236457.3125	901743.1251
497	234463.3125	901261.6251		1512	236457.9531	901452.4375
498	234466.4219	899511.5625		1513	236458.6251	900413.3125
499	234473.4063	899531.8125		1514	236459.9375	900621.0625
500	234477.0157	900427.1875		1515	236460.0781	901713.3125
501	234487.9845	899920.3125		1516	236460.3751	899828.2501
502	234499.1875	900368.7501		1517	236461.5625	901536.1251
503	234503.6875	901992.8751		1518	236462.7501	902016.4375
504	234515.5157	901517.9375		1519	236464.3125	901667.8125

505	234516.3595	900323.2501		1520	236465.2031	901867.0001
506	234532.4845	900281.4375		1521	236466.4845	901644.3751
507	234532.5157	899450.6251		1522	236469.0937	901336.7501
508	234534.9687	899914.3125		1523	236469.6875	900112.6875
509	234536.4687	899605.0001		1524	236472.8595	899866.3751
510	234538.0781	901904.2501		1525	236473.0781	901576.8125
511	234548.4531	902063.3125		1526	236473.3751	901026.7501
512	234550.1251	900234.8751		1527	236473.4375	900647.4375
513	234551.2031	899899.7501		1528	236474.7501	901550.1875
514	234555.4687	899945.6251		1529	236475.6719	901173.5625
515	234564.9375	900195.7501		1530	236475.7969	901801.7501
516	234568.8125	902136.5001		1531	236480.0469	900797.3125
517	234580.9687	899627.6251		1532	236480.0625	900214.9375
518	234581.1719	900150.3751		1533	236481.7969	901532.1875
519	234581.9375	902210.6251		1534	236482.1095	900871.5625
520	234582.2657	901599.1875		1535	236482.5781	900888.5625
521	234582.4375	901340.4375		1536	236482.7345	899392.8125
522	234585.9375	899512.5001		1537	236483.0625	900398.6875
523	234586.0469	901293.0001		1538	236483.1719	899470.0001
524	234586.5781	901307.5625		1539	236483.5937	900272.2501
525	234591.8437	901978.6875		1540	236485.8281	901715.3125
526	234592.7501	902287.6875		1541	236488.1095	901294.0625
527	234593.8751	899785.6875		1542	236490.8437	901026.2501
528	234595.2345	901670.2501		1543	236490.8907	901443.1251
529	234597.9531	900108.8125		1544	236490.9063	901668.5001
530	234599.2657	899771.2501		1545	236491.4063	900331.2501
531	234599.9687	901305.1251		1546	236492.6251	900312.3125
532	234603.7813	899686.5625		1547	236492.8751	901975.6251
533	234604.7813	899786.0001		1548	236493.7187	900322.7501
534	234605.1563	902364.6251		1549	236494.4531	900329.6875
535	234605.3281	902051.5625		1550	236494.5937	901277.8125
536	234606.9219	901745.1251		1551	236494.6875	900332.6251
537	234610.6563	899600.5625		1552	236494.8125	899987.6875
538	234611.2501	901320.1875		1553	236495.9531	901079.3751
539	234615.5937	901336.8751		1554	236496.0313	900342.8751
540	234616.1251	900063.8125		1555	236497.9845	900390.3751
541	234618.1875	902443.2501		1556	236498.2657	900983.4375
542	234618.2031	902127.3751		1557	236500.0937	901261.5625
543	234618.5625	901820.6251		1558	236503.2969	901327.8125
544	234621.6719	899713.0001		1559	236503.6251	900141.0001
545	234624.4063	901353.5625		1560	236506.7031	901364.5625
546	234630.0313	901297.5625		1561	236507.4375	899887.0625
547	234630.6875	901895.5001		1562	236509.0157	901025.5001
548	234631.0157	900020.5625		1563	236512.0313	900514.9375

549	234633.0937	899699.0625		1564	236514.4845	902059.0625
550	234635.8595	899511.4375		1565	236515.9375	901753.1875
551	234640.5469	899575.7501		1566	236517.0157	899647.0625
552	234640.9531	899689.4375		1567	236517.7657	901190.5001
553	234644.3281	901970.5625		1568	236518.6095	901317.5001
554	234645.5781	899975.5625		1569	236520.2969	900201.7501
555	234654.3125	900510.3125		1570	236521.0001	901374.6875
556	234657.2031	899660.5625		1571	236521.0781	901647.4375
557	234657.4219	902042.5625		1572	236521.1095	900982.9375
558	234659.1251	900493.6875		1573	236522.4687	901609.0625
559	234664.1875	899485.2501		1574	236523.5157	901580.6251
560	234666.3751	901278.0625		1575	236524.8595	899668.0001
561	234669.3595	899550.0625		1576	236526.6407	899949.5625
562	234671.8125	902117.5001		1577	236528.7813	900482.8125
563	234675.4687	900436.6875		1578	236529.5937	900800.8751
564	234675.9063	899884.3751		1579	236530.2657	900088.7501
565	234681.5469	901309.0001		1580	236530.8751	901770.2501
566	234683.5781	902194.3751		1581	236532.3907	900936.9375
567	234684.8437	901331.2501		1582	236532.4531	901890.8751
568	234686.3437	901349.5625		1583	236532.4531	901945.6251
569	234689.3751	901395.1875		1584	236532.6719	901850.6251
570	234690.3751	899774.0625		1585	236532.9531	901799.5625
571	234691.7969	899632.7501		1586	236535.6251	901298.5625
572	234693.1095	900387.7501		1587	236536.7657	899953.3125
573	234693.4375	901257.2501		1588	236538.8281	902018.8125
574	234696.1563	902267.9375		1589	236541.1875	900898.8125
575	234698.4219	901457.6251		1590	236541.4687	900351.1251
576	234702.9531	899809.1251		1591	236542.0781	902044.0001
577	234703.5625	899789.6875		1592	236544.5469	900654.5001
578	234703.7657	901494.6875		1593	236545.2501	901281.5001
579	234705.7187	899579.4375		1594	236545.5937	900365.1251
580	234707.1875	902347.9375		1595	236547.4531	901263.6251
581	234708.0313	899795.0001		1596	236547.8751	901601.3751
582	234708.4375	900346.0625		1597	236548.6407	900883.0001
583	234709.4687	899774.6251		1598	236550.4531	901483.6251
584	234709.8595	899789.9375		1599	236550.6719	901788.7501
585	234712.5469	899782.4375		1600	236553.7657	901623.3125
586	234716.6563	901577.0001		1601	236554.1095	900803.3751
587	234719.5625	899790.2501		1602	236555.3125	900360.9375
588	234720.6875	899741.7501		1603	236556.4531	900542.0625
589	234721.7345	902424.0001		1604	236557.0469	901648.1875
590	234722.4845	899499.0625		1605	236557.2187	901712.4375
591	234723.6719	901328.6251		1606	236560.2501	900358.7501
592	234724.3281	900304.1251		1607	236562.9687	900183.3751

593	234725.5157	901344.8751		1608	236567.0313	901512.3751
594	234731.1251	901651.5001		1609	236568.3907	900340.7501
595	234735.0001	901479.9375		1610	236568.7345	901684.6875
596	234735.7345	901456.0625		1611	236568.7813	900355.0625
597	234736.0469	899831.7501		1612	236568.8281	900488.9375
598	234740.9063	900260.0625		1613	236570.1719	901664.5625
599	234741.7969	901720.8125		1614	236570.8437	900354.1875
600	234753.6095	899536.8751		1615	236572.7969	901219.6875
601	234755.5625	901798.3125		1616	236574.9219	900447.3125
602	234757.9845	900215.0625		1617	236576.4375	901100.5625
603	234761.2969	899776.4375		1618	236578.7501	901768.1251
604	234761.4687	902434.2501		1619	236580.3595	900388.8751
605	234761.8751	899575.0625		1620	236582.2031	901541.6251
606	234768.0937	901870.6875		1621	236587.7657	901300.0625
607	234772.9531	902444.2501		1622	236588.8907	901045.8751
608	234773.8281	900172.5001		1623	236589.7813	900236.2501
609	234774.9375	899562.8125		1624	236589.8281	902061.3751
610	234783.2501	901947.7501		1625	236591.7501	902016.8125
611	234789.1563	900129.8125		1626	236593.5469	901758.4375
612	234791.3907	899785.6875		1627	236597.9845	900807.9375
613	234796.4219	902021.2501		1628	236598.4845	900391.5001
614	234803.4219	899594.1875		1629	236599.5937	901445.4375
615	234805.1095	900085.6875		1630	236600.6563	901581.0625
616	234805.9845	899794.0001		1631	236602.4531	900560.6251
617	234809.7031	902094.3751		1632	236603.2813	901811.8125
618	234821.5313	900039.4375		1633	236604.7813	900669.1875
619	234821.8437	901715.4375		1634	236606.7657	901101.9375
620	234822.1875	902173.1875		1635	236612.2345	900338.1875
621	234834.1407	902246.3125		1636	236613.3125	900961.6251
622	234835.4687	900560.7501		1637	236613.5625	900290.8751
623	234838.7187	899456.2501		1638	236613.7501	901434.5625
624	234847.1563	902323.9375		1639	236614.7031	901614.5001
625	234851.2031	899602.8751		1640	236617.8437	901640.9375
626	234854.1719	900502.6875		1641	236618.1095	901304.8751
627	234857.8281	899951.2501		1642	236618.6875	901659.5625
628	234864.1719	899484.2501		1643	236619.2969	901282.5625
629	234865.4531	901780.3751		1644	236619.4063	901844.1875
630	234870.5001	900453.1251		1645	236619.8437	901677.8751
631	234882.9687	900613.0001		1646	236621.0313	901710.6875
632	234884.3907	900411.8125		1647	236621.5469	901736.1875
633	234895.7969	899456.8125		1648	236626.2501	901561.7501
634	234896.7187	899510.4375		1649	236627.0001	900931.2501
635	234897.9845	899841.1875		1650	236628.3437	901333.5625
636	234898.1875	899561.7501		1651	236629.7031	901479.8125

637	234901.5625	900370.6875		1652	236630.7031	900331.4375
638	234902.2031	902312.5001		1653	236634.0001	901055.6875
639	234913.9219	899798.7501		1654	236635.2187	900910.4375
640	234917.8907	902332.3125		1655	236637.0313	901354.4375
641	234919.8437	900324.7501		1656	236638.8281	901358.7501
642	234919.9531	899782.6875		1657	236639.2345	901603.5625
643	234925.9845	902344.3751		1658	236641.5313	901365.1251
644	234936.5157	900278.0001		1659	236642.5625	901364.3125
645	234944.9063	899988.5001		1660	236643.1875	901987.4375
646	234946.4375	900624.5625		1661	236643.5469	900497.3751
647	234949.8751	899516.3751		1662	236648.9375	901506.0625
648	234949.9219	900239.8125		1663	236651.5781	901400.7501
649	234958.5625	902293.4375		1664	236651.9531	901806.2501
650	234960.7187	901920.3125		1665	236653.2813	901884.2501
651	234965.9531	900193.6251		1666	236653.5001	901084.8125
652	234970.8437	900633.8125		1667	236653.7501	900922.3751
653	234977.4531	901909.9375		1668	236654.7501	901858.1251
654	234982.0469	900151.3125		1669	236655.3437	901389.9375
655	234985.5157	899873.6251		1670	236655.4687	900872.0625
656	234998.1251	900644.6875		1671	236656.5937	901461.1251
657	234998.8437	900107.1251		1672	236658.6563	900845.1875
658	234999.3125	899571.6251		1673	236659.8437	901823.4375
659	235009.7031	900623.4375		1674	236661.6095	901872.8751
660	235010.7345	899567.3125		1675	236662.6875	901457.7501
661	235011.6095	899539.6251		1676	236663.2813	900959.6875
662	235029.2969	900567.8751		1677	236663.3595	901927.8125
663	235033.9375	900016.5625		1678	236665.7345	901529.3125
664	235041.3125	899457.5625		1679	236667.5313	901840.0001
665	235048.2657	900518.5625		1680	236668.1875	900712.8125
666	235056.1095	902021.0001		1681	236671.5157	901311.5625
667	235063.7813	900479.0001		1682	236672.7031	901490.1251
668	235070.9375	899913.8751		1683	236672.7969	900856.8751
669	235078.3281	900434.7501		1684	236673.5937	902033.6875
670	235090.5781	899859.0625		1685	236674.5469	900913.0001
671	235095.9375	900389.5001		1686	236675.7657	900579.8751
672	235102.8907	899828.4375		1687	236676.7187	901919.0625
673	235107.0157	899764.1251		1688	236677.6095	901862.0001
674	235110.8437	899805.3125		1689	236679.0001	901111.1251
675	235111.5937	900340.1251		1690	236681.0313	901333.9375
676	235116.2187	899791.7501		1691	236682.1563	901654.2501
677	235120.4845	899805.6251		1692	236682.9845	901316.6875
678	235126.3125	900302.5625		1693	236683.1407	901958.3125
679	235126.4687	899766.8751		1694	236685.3281	901618.9375
680	235136.3907	900688.3751		1695	236687.7501	901560.0001

681	235139.2813	899728.5001		1696	236689.2501	901065.0001
682	235141.3751	900714.4375		1697	236690.8595	901318.8751
683	235142.7813	900260.5001		1698	236693.9687	900423.3751
684	235144.7813	900695.2501		1699	236693.9845	900557.1251
685	235149.4063	900748.7501		1700	236695.2031	901766.1875
686	235149.5157	902166.0001		1701	236697.2813	901366.3751
687	235153.3751	900678.3125		1702	236700.2345	901951.0625
688	235156.3751	902175.6251		1703	236704.2187	901114.3125
689	235158.7657	900216.9375		1704	236706.7031	901802.6251
690	235161.5625	902157.8125		1705	236709.7345	901293.2501
691	235163.8595	899848.7501		1706	236711.9687	900891.7501
692	235163.9531	900682.4375		1707	236712.4845	901003.4375
693	235168.6563	902168.0001		1708	236713.8281	901462.8125
694	235174.7345	899631.2501		1709	236714.8125	901069.1875
695	235174.9687	900716.3125		1710	236715.5469	901593.9375
696	235175.2345	900170.9375		1711	236715.7345	901869.6875
697	235176.8281	900622.6875		1712	236716.4845	901940.0625
698	235182.8125	900741.6875		1713	236718.1875	900589.0625
699	235183.5937	900809.6875		1714	236719.4375	901789.9375
700	235183.7345	900690.2501		1715	236723.7031	901509.1875
701	235184.4687	899508.3751		1716	236725.2345	900969.6875
702	235185.8125	899600.8751		1717	236727.3907	901424.3125
703	235194.4219	900630.1875		1718	236730.3437	901681.0001
704	235195.5157	901283.1251		1719	236731.7187	901708.4375
705	235195.5157	901299.3125		1720	236733.2187	902009.8125
706	235195.6875	900116.8125		1721	236733.7657	900953.2501
707	235197.8907	899792.1875		1722	236734.3125	901496.8125
708	235203.2969	899730.0001		1723	236735.9063	900452.3751
709	235203.7813	901339.8751		1724	236739.4375	900497.6251
710	235204.5625	900841.0625		1725	236740.9531	901447.8125
711	235204.8125	899550.6251		1726	236741.8125	901723.6251
712	235206.6407	899810.0625		1727	236743.1719	901925.3751
713	235207.9845	899803.1875		1728	236743.3125	900622.1875
714	235209.3125	900078.5001		1729	236744.2187	900490.1875
715	235210.7969	899806.2501		1730	236745.1095	901851.1251
716	235212.3595	900894.6251		1731	236745.4531	900930.8125
717	235214.0937	899810.0001		1732	236746.8125	901537.3751
718	235218.7657	900638.8125		1733	236746.9063	901632.2501
719	235220.0313	899509.0001		1734	236747.2813	901574.3751
720	235221.1251	902117.6875		1735	236748.1095	901444.1875
721	235228.3751	901278.0001		1736	236748.9375	901531.4375
722	235228.6719	902126.9375		1737	236750.1095	900480.9375
723	235228.8907	901296.2501		1738	236751.7031	901075.2501
724	235231.7187	899652.6251		1739	236752.5937	900770.9375

725	235231.9063	900849.1875		1740	236753.5313	901518.6251
726	235235.4375	900708.2501		1741	236754.4219	901711.0001
727	235236.7187	900588.6875		1742	236755.2345	901339.8751
728	235239.3281	901011.1251		1743	236758.0781	900468.5001
729	235240.5313	900825.8125		1744	236760.6563	901242.7501
730	235242.6251	899916.5001		1745	236760.7969	900868.0001
731	235243.9063	899981.2501		1746	236762.1719	900463.1875
732	235245.3751	900809.8751		1747	236765.4063	901208.3125
733	235251.2031	900548.7501		1748	236765.7031	901249.0625
734	235252.7813	901082.7501		1749	236766.9531	901547.3751
735	235254.5157	901140.0625		1750	236768.5313	901021.7501
736	235255.3907	901112.5625		1751	236770.7187	901556.6875
737	235255.7187	899740.1251		1752	236771.0781	901464.6251
738	235257.4531	900652.8125		1753	236771.3751	901305.0625
739	235260.0157	900857.3125		1754	236771.7501	901256.1875
740	235262.0001	901179.6251		1755	236773.3437	901530.0625
741	235262.5937	901202.3751		1756	236773.4063	901490.1875
742	235264.4063	899924.6251		1757	236774.3437	900852.5625
743	235264.5625	902174.3125		1758	236775.0469	901259.8751
744	235266.3437	900834.9375		1759	236775.0625	901149.3751
745	235271.5781	900818.1875		1760	236775.5313	900526.8125
746	235272.0001	900502.5625		1761	236776.1875	900449.0001
747	235272.1251	902343.4375		1762	236778.7501	901124.4375
748	235274.9531	902088.5625		1763	236779.6719	901889.8751
749	235275.3437	901383.0001		1764	236782.3595	901080.2501
750	235276.2657	901011.6251		1765	236782.6407	901268.6251
751	235277.5937	901430.6875		1766	236784.8437	901605.2501
752	235279.7657	899883.5001		1767	236786.1407	901987.7501
753	235279.8907	899670.6251		1768	236787.8595	901689.5625
754	235286.2501	901293.3125		1769	236790.0469	901498.9375
755	235286.8125	900770.0625		1770	236791.3281	900796.8751
756	235291.6563	900458.9375		1771	236794.3437	902094.5625
757	235296.0157	899840.3125		1772	236795.2187	901600.7501
758	235299.4531	900729.5001		1773	236795.9375	901283.8125
759	235303.7969	902234.5625		1774	236798.8125	900543.3125
760	235304.2031	901322.6875		1775	236799.6563	901083.2501
761	235305.3437	900713.1251		1776	236803.7187	901714.6875
762	235306.6407	899812.7501		1777	236803.9531	900982.8751
763	235307.6251	900415.4375		1778	236808.1251	901678.8751
764	235309.1875	899806.1875		1779	236809.3437	900897.2501
765	235311.5937	901110.9375		1780	236809.3437	901023.8125
766	235314.4531	899792.5625		1781	236809.4687	900492.5001
767	235316.7657	901178.7501		1782	236810.7031	901589.7501
768	235318.6251	900674.5625		1783	236811.3907	900810.3125

769	235321.8907	900375.8751		1784	236812.5157	901586.9375
770	235323.3595	901200.3751		1785	236813.6251	901421.5625
771	235327.8125	901292.2501		1786	236814.4845	900713.9375
772	235332.1875	901416.5001		1787	236817.9219	901540.1875
773	235332.8751	900975.5001		1788	236819.2031	901742.3125
774	235336.6251	901012.8751		1789	236821.3595	901509.7501
775	235338.2187	900330.5625		1790	236821.9219	900920.5001
776	235339.3281	900938.3125		1791	236822.7345	901219.9375
777	235339.9063	902455.9375		1792	236822.9531	902082.6251
778	235340.5157	900233.9375		1793	236823.6407	901489.6251
779	235347.9687	902296.7501		1794	236825.7031	901772.1875
780	235349.2501	901246.8125		1795	236826.4063	901970.1251
781	235351.1407	901220.0001		1796	236829.7501	900795.1875
782	235351.5157	899696.6251		1797	236829.9063	901602.5625
783	235352.5001	900132.0001		1798	236829.9845	901883.8751
784	235353.5781	900290.7501		1799	236831.5625	901799.5625
785	235358.2969	901089.3751		1800	236833.9531	901405.9375
786	235361.2187	902194.1875		1801	236835.4219	902077.1251
787	235362.7345	899664.8751		1802	236836.3595	901824.3125
788	235363.5469	901013.8125		1803	236840.4219	901544.2501
789	235364.5469	901424.0625		1804	236840.9845	900200.3751
790	235366.4063	900889.1251		1805	236841.3595	901850.8125
791	235366.5625	901315.5625		1806	236844.7187	900958.8751
792	235369.5469	900249.3125		1807	236845.5781	901877.5625
793	235373.5469	902000.0001		1808	236845.6719	900555.6251
794	235373.6095	900239.7501		1809	236847.8751	901533.8751
795	235375.3437	901291.1875		1810	236850.9531	900626.8125
796	235379.0625	900857.0001		1811	236853.4375	901618.1251
797	235380.3751	901308.2501		1812	236854.6251	901263.3125
798	235380.8125	899614.8751		1813	236856.3125	901432.6875
799	235383.4375	901290.1875		1814	236858.5469	900761.9375
800	235388.5157	900034.7501		1815	236859.3125	901239.3125
801	235390.0001	902210.8751		1816	236859.7345	901517.3751
802	235390.4063	900191.3751		1817	236860.8751	900868.7501
803	235395.3437	899572.5001		1818	236862.7345	901953.1251
804	235400.1719	900899.9375		1819	236863.1719	901383.5001
805	235400.5001	901014.4375		1820	236865.6251	899491.3751
806	235402.2969	900803.4375		1821	236866.2031	901550.6875
807	235404.1407	900152.0625		1822	236870.7969	900900.3125
808	235406.0157	902000.0001		1823	236872.7657	900643.5001
809	235407.2813	901607.8751		1824	236872.9845	901096.6875
810	235410.8595	901090.8751		1825	236878.1563	901645.5625
811	235412.4375	901255.4375		1826	236881.3907	901101.6875
812	235413.2813	899527.0625		1827	236882.4063	901002.0001

813	235415.2345	902227.5625		1828	236883.3751	900802.9375
814	235421.6095	901178.1251		1829	236883.7187	900856.6251
815	235422.1719	901037.0001		1830	236885.3907	901642.1251
816	235423.2969	899822.7501		1831	236885.8437	901104.3751
817	235423.5469	900753.5625		1832	236886.8751	901035.9375
818	235424.7187	899632.0001		1833	236890.2187	900936.5625
819	235425.4375	901882.1251		1834	236891.0469	901573.6251
820	235427.4063	901275.2501		1835	236891.0937	901897.7501
821	235428.1875	901263.8125		1836	236891.4845	900306.8125
822	235428.6719	901014.8125		1837	236893.0469	900772.9375
823	235430.0937	899487.7501		1838	236893.6251	901004.2501
824	235436.1875	900908.9375		1839	236894.3751	901931.3125
825	235437.0313	900050.5625		1840	236894.8907	901037.1875
826	235438.0001	900815.0001		1841	236896.6719	902041.7501
827	235438.9531	900714.0001		1842	236897.7813	900617.7501
828	235439.1719	901699.9375		1843	236898.6875	901564.5625
829	235440.6251	901629.5001		1844	236900.3437	902213.9375
830	235442.9845	901951.5001		1845	236901.2345	901792.0625
831	235443.2345	899508.3751		1846	236901.2969	901006.0001
832	235447.2345	901125.8125		1847	236903.5781	901019.5001
833	235448.9219	901093.1251		1848	236905.9063	900886.1251
834	235449.9531	901179.5625		1849	236906.1251	901007.2501
835	235452.1875	901961.1875		1850	236908.5001	901337.7501
836	235452.2813	899996.3751		1851	236908.5937	901007.9375
837	235452.5937	901669.4375		1852	236909.1095	899467.3125
838	235453.3437	900779.3125		1853	236909.1563	902040.1251
839	235454.7969	901145.2501		1854	236909.4845	901946.6251
840	235460.0313	899962.0001		1855	236910.5781	900926.3751
841	235463.2969	899738.2501		1856	236911.4845	901113.3751
842	235466.4375	899837.3125		1857	236911.5157	901915.0001
843	235467.0001	900915.0001		1858	236912.4845	901541.3751
844	235469.4375	900825.4375		1859	236912.6719	901464.0625
845	235470.0937	901264.1251		1860	236916.3751	901597.0001
846	235470.4063	901237.8751		1861	236918.5001	900922.4375
847	235473.7031	901130.6875		1862	236920.5937	901702.3751
848	235474.1251	900724.7501		1863	236920.7813	901681.4375
849	235474.9531	899895.5001		1864	236921.6719	901633.0625
850	235478.5625	901094.5625		1865	236922.3751	901626.1251
851	235479.9845	899879.1875		1866	236922.5469	900883.8751
852	235480.5781	901181.1251		1867	236923.5469	901116.0001
853	235482.0469	900870.5625		1868	236924.6719	900921.5001
854	235484.9687	899860.1251		1869	236925.0625	901741.3751
855	235485.0781	901144.1251		1870	236925.3437	900876.3751
856	235487.4531	902114.3125		1871	236925.6719	899762.0625

857	235493.3281	899828.3125		1872	236932.2187	900829.5001
858	235495.0001	900275.2501		1873	236932.5781	900882.5001
859	235495.2813	901275.8125		1874	236932.8125	901014.1875
860	235496.3125	899820.5625		1875	236932.9687	901764.1875
861	235496.3125	900833.8751		1876	236933.0937	901498.0625
862	235499.6407	901133.3751		1877	236933.1095	900582.5001
863	235501.2501	901095.7501		1878	236933.4687	900582.5625
864	235501.3437	901017.1251		1879	236936.5469	899747.8751
865	235503.0469	899806.2501		1880	236936.5625	900881.9375
866	235503.5157	900647.7501		1881	236938.4375	901887.9375
867	235504.4375	899658.5001		1882	236938.7969	900824.4375
868	235506.9531	899976.1875		1883	236939.0937	900921.7501
869	235511.7969	899791.5001		1884	236939.4845	900821.9375
870	235514.0469	899476.5625		1885	236940.1719	901698.3125
871	235515.2345	900211.8751		1886	236940.3907	901619.1875
872	235520.6095	900923.1875		1887	236940.4219	901782.3125
873	235520.6563	901412.9375		1888	236944.9375	900862.1875
874	235523.8125	899926.0001		1889	236945.2969	901017.2501
875	235524.5625	900149.5001		1890	236946.0313	900789.7501
876	235525.0937	899498.7501		1891	236947.4063	901625.6251
877	235525.2501	901265.3751		1892	236949.9531	900032.1251
878	235525.7813	899713.0625		1893	236951.8437	900923.1875
879	235526.4845	901276.7501		1894	236952.1251	901528.8125
880	235529.8751	901017.8751		1895	236953.3437	900813.2501
881	235531.2969	901401.8125		1896	236954.5313	900632.7501
882	235531.5157	901182.6875		1897	236955.1719	899650.8125
883	235534.5001	901384.1875		1898	236955.5469	901864.3125
884	235536.2345	901097.5001		1899	236955.8595	900879.3125
885	235536.8751	900088.7501		1900	236955.9219	901817.0001
886	235537.7345	899817.0625		1901	236957.1875	900570.4375
887	235538.3281	902287.1251		1902	236958.0469	901738.4375
888	235540.2031	901072.0625		1903	236958.4687	900792.8125
889	235540.9845	901041.6875		1904	236961.4845	900793.4375
890	235541.7969	901898.5001		1905	236962.2813	901840.3751
891	235542.4687	899888.4375		1906	236964.9063	901966.3751
892	235543.1719	901868.1251		1907	236965.7969	900921.8751
893	235544.1563	900057.5625		1908	236971.1251	900734.2501
894	235544.2657	899754.2501		1909	236971.5781	899890.5001
895	235548.0937	900926.5625		1910	236973.9845	900799.3751
896	235552.6719	900021.0625		1911	236974.0313	899972.5001
897	235552.7657	899842.1875		1912	236974.1875	900672.1875
898	235555.4531	901880.2501		1913	236974.5625	901751.6875
899	235557.0625	900893.0001		1914	236974.7969	902177.8125
900	235557.1875	900550.1875		1915	236975.0781	900800.0625

901	235560.0625	899990.6251		1916	236975.2345	901734.4375
902	235562.2031	902477.5001		1917	236975.4375	901729.0625
903	235563.0937	899828.6251		1918	236976.9219	901690.6251
904	235564.1563	902475.6875		1919	236977.0469	900596.3125
905	235564.9845	901863.8125		1920	236977.6563	901653.5625
906	235566.8437	900854.0625		1921	236977.7187	900876.3751
907	235567.0157	899785.9375		1922	236978.3907	901616.6251
908	235567.3437	902472.6875		1923	236978.6251	901604.3751
909	235567.4063	901267.3125		1924	236979.5313	899432.5001
910	235568.0625	902476.3125		1925	236980.8751	901558.5001
911	235568.4375	900197.1251		1926	236981.7187	901402.6251
912	235569.1875	901233.3751		1927	236982.6251	901523.1251
913	235569.5937	900518.6251		1928	236984.5157	900807.8751
914	235574.9375	902474.8751		1929	236985.5157	899700.7501
915	235578.2969	899932.3125		1930	236986.1563	900612.1251
916	235578.7345	899797.8751		1931	236986.7501	901439.9375
917	235581.2969	900106.3125		1932	236988.0313	899999.5625
918	235583.0625	901855.9375		1933	236989.3125	900812.6875
919	235583.8751	899914.3125		1934	236993.8125	900105.2501
920	235583.9063	901133.9375		1935	236995.4219	901746.5001
921	235584.7657	901916.7501		1936	237000.0001	899386.0625
922	235589.2187	900071.0625		1937	237000.1251	899629.5625
923	235592.4063	899810.3125		1938	237002.6719	900659.3125
924	235594.1563	900862.2501		1939	237011.1719	900722.5625
925	235596.4845	901363.5001		1940	237014.5157	900820.6251
926	235596.6875	901386.0625		1941	237018.0001	899974.4375
927	235597.5937	900033.8751		1942	237025.1875	899768.2501
928	235598.7031	900229.1251		1943	237026.9219	899677.3751
929	235599.4531	901279.0625		1944	237028.9375	899774.6875
930	235600.5625	901268.8751		1945	237030.5001	900159.5001
931	235601.1875	900002.4375		1946	237031.3595	899769.6875
932	235601.2031	901860.5625		1947	237034.8437	900586.3751
933	235602.6563	899881.3125		1948	237036.0937	899787.5625
934	235603.2501	900826.6251		1949	237037.2969	899771.0625
935	235604.8125	901184.1875		1950	237040.4375	900011.1875
936	235605.0001	902468.6875		1951	237044.8437	902122.0625
937	235606.5313	900398.3751		1952	237046.9687	900637.0625
938	235607.9219	900809.0001		1953	237058.4219	901868.1875
939	235608.1875	901133.5001		1954	237060.9219	900136.3751
940	235610.3281	899973.5625		1955	237063.6251	899837.2501
941	235610.7657	901099.5001		1956	237074.2501	901961.0625
942	235611.8907	899928.4375		1957	237076.7969	901745.0625
943	235613.3437	901019.8751		1958	237087.3907	900181.7501
944	235613.8437	901069.0625		1959	237097.5625	901529.8125

945	235613.9063	901935.8751		1960	237100.0157	900854.3751
946	235614.3437	901043.7501		1961	237100.0469	899901.0001
947	235616.7031	899732.1875		1962	237101.2187	901602.7501
948	235619.1875	900768.5001		1963	237111.9531	900224.6875
949	235622.9687	900349.5625		1964	237118.0469	901481.6875
950	235624.2031	900189.3125		1965	237118.1719	900235.5001
951	235625.2031	901588.1251		1966	237120.8751	899625.2501
952	235625.6251	901362.6251		1967	237120.9219	899938.5625
953	235630.0625	901947.2501		1968	237124.6719	901734.2501
954	235631.2813	900123.9375		1969	237124.8751	901519.4375
955	235632.4375	899845.2501		1970	237133.5001	899959.9375
956	235633.2657	899905.1251		1971	237140.6875	900274.8751
957	235635.0625	901892.8125		1972	237150.8751	899701.7501
958	235635.8125	900312.3125		1973	237157.7031	900121.7501
959	235639.6407	901839.3125		1974	237158.3907	900004.1875
960	235641.5313	901777.4375		1975	237166.3125	899693.6251
961	235643.6407	900046.3751		1976	237174.8595	900038.4375
962	235649.2657	899855.5001		1977	237182.4219	900165.8125
963	235650.2187	900823.5625		1978	237183.6875	900248.6875
964	235652.2345	900179.7501		1979	237202.9063	900083.1875
965	235654.0625	900237.0001		1980	237203.0937	900382.1875
966	235655.6251	901921.8125		1981	237205.0781	900495.5001
967	235659.5781	901217.3751		1982	237205.7187	900284.3751
968	235659.8907	899800.8751		1983	237207.4687	901174.1875
969	235661.8437	900020.5001		1984	237210.2187	900392.3125
970	235662.7031	899491.4375		1985	237212.0157	900141.1251
971	235665.8281	901834.0001		1986	237213.6875	899808.5625
972	235666.8751	901670.2501		1987	237216.5313	899825.6875
973	235668.3751	900845.6251		1988	237223.9687	900065.2501
974	235669.9063	899513.3751		1989	237235.6563	899894.0001
975	235670.7969	901524.6875		1990	237239.3125	900267.3125
976	235674.8125	899938.3125		1991	237242.9687	900345.7501
977	235674.8437	901390.6251		1992	237248.5157	900112.9375
978	235675.4531	901411.4375		1993	237249.9375	900469.9375
979	235676.4375	901278.2501		1994	237250.7187	899673.8125
980	235676.9063	901272.9375		1995	237257.1407	899978.2501
981	235676.9375	901970.8125		1996	237258.8595	900035.5001
982	235678.9687	901218.2501		1997	237263.0781	899762.8751
983	235680.1563	901186.7501		1998	237264.8437	900312.8751
984	235681.0625	899533.6251		1999	237268.5157	899694.3751
985	235681.6563	899994.0625		2000	237279.6563	899930.8125
986	235682.1563	900310.8751		2001	237288.5469	899925.7501
987	235683.6095	901103.5625		2002	237299.1095	900290.1251
988	235683.7187	901020.6875		2003	237303.3281	900212.1875

989	235685.1251	901074.1875		2004	237311.3125	899994.3125
990	235686.1875	900142.2501		2005	237316.0625	899910.0625
991	235688.5001	900997.3125		2006	237318.0937	899751.3125
992	235700.8751	900061.7501		2007	237327.8907	900424.0001
993	235703.5781	901525.4375		2008	237334.5313	900267.5625
994	235708.4063	899897.9375		2009	237343.8751	899443.0625
995	235709.6563	900004.3125		2010	237346.6563	900184.8751
996	235710.0157	902240.1251		2011	237364.2031	899804.2501
997	235710.1407	900246.5625		2012	237365.5469	900215.1251
998	235714.9687	899969.6875		2013	237397.3125	899851.1251
999	235717.0313	902449.1875		2014	237410.4375	899383.6875
1000	235719.3595	901618.1875		2015	237416.2031	899853.3751
1001	235720.4219	901279.3125		2016	237420.3751	899836.0625
1002	235720.8907	901275.5001		2017	237428.3751	899830.8751
1003	235721.7969	901188.7501		2018	237430.3281	899938.1875
1004	235723.9531	901104.0625		2019	237440.5001	900233.8751
1005	235727.7345	901074.8125		2020	237441.9219	899838.5001
1006	235729.8751	900959.5625		2021	237444.9687	900611.0001
1007	235735.3907	901815.8125		2022	237449.0157	899832.9375
1008	235737.1095	901759.0001		2023	237449.4845	900244.9375
1009	235740.7969	902419.5001		2024	237451.0625	899816.6251
1010	235742.8125	900909.3125		2025	237458.3907	899825.5625
1011	235748.7969	899614.1251		2026	237502.0313	899959.6875
1012	235749.6563	901526.7501		2027	237537.8595	902219.5625
1013	235750.7813	901411.4375		2028	237578.4531	902208.0625
1014	235752.9845	901278.5625		2029	237595.7657	899603.3751
				2030	237960.4531	899689.5001

Appendix C: Matlab function lists

All Matlab source files are stored in a folder 'AdhocNet'.

Its subfolder 'City map model' contains Matlab functioning files of EP (Evolutionary Programming) technique.

The subfolder 'Stochastic Geometry' contains Matlab functioning files of SG technique.

The subfolder 'RWP model' contains Matlab functioning files of EP technique, in comparison with SG technique.

The subfolder 'BER analysis' contains Matlab functioning files to generate BER from distance.

All functions are included in list below:

ID	File name	File information
Subfolder: City map model		
1	Test_2_Boston_City_all_roads_NO1_road_ini	Read in data sets from source file 'Node_XY' and create a graph of nodes and edges
2	Test_3_roadfilter_area_1	Generates randomly connected pair of nodes as input of function 4
3	Test_3_Boston_City_map_ini_2	Calculate distance matrix
4	Test_3_City_EP_Dynamic_mode_3	Optimize the connections requested by function 2, with fitness defined as signal quality
5	Test_3_Sensor_BER_No_4	Optimize the connections requested by function 2, with fitness defined as sensor number
6	Test_3_Multi_Obj_EP_5	Define multiple objective fitness function and optimize the connections requested by function

		2
7	Test_3_Spatial_node_pair_GA _6	Choose pairs of nodes according to their spatial distance distribution and optimize with EP, for comparison with SG results
8	Test_3_Boston_City_AODV_7	(Obsoleted) Use AODV as path routing algorithm, compare results with EP
9	Test_4_Area_2_ini_A	(Obsoleted)Define area 2 in Boston city, optimize the connection pairs with EP
10	Test_4_Spatial_node_pair_GA _B	(Obsoleted) Define area 2 in Boston city, choose pairs of nodes according to their spatial distance distribution and optimize them with EP
Subfolder: Stochastic Geometry		
11	Junction_density_distributio n_1	Calculate the street junction density distribution and find the value of λ
12	Virtual_Poisson_graph_2	Apply Poisson point process to each subarea according to λ and create a virtual Poisson graph
13	Data_loss_rate_3	Apply equation 5.1 and calculate data loss rate of each subarea
14	Node_selection_4	Define central nodes of each subarea
15	Transition_matrix_5	Apply Markov Chain on the central nodes set and calculate state transition matrix
Subfolder: RWP model		
16	Test1_5_NO1_node_power	(Obsoleted) Calculate node transmit energy
17	Test1_5_NO2_node_connecti on_power	(Obsoleted) Calculate channel energy consumption
18	Test1_5_NO3_node_power_E P	(Obsoleted) Optimize connection routes according to their power consumption status
19	Test1_5_NO4_EP_fitness_anal ysis	Apply EP in optimization of
20	Test2_20_NO1_dense_net_ini	Initiate a random network of 50 nodes (simulation results in chapter 2)
21	Test2_20_NO2_dense_net_co nnection_power	(Obsoleted) Calculate channel energy consumption
22	Test2_20_NO3_dense_net_no de_power	(Obsoleted) Calculate node transmit power
23	Test2_20_NO4_dense_net_EP _50_iteration	Optimizes connections with EP in 50 iterations
Subfolder: BER analysis		

24	BER_Distance_fading_1	Apply Rayleigh fading coefficient to calculate distance fading
25	BER_Street_movement_2	Generate pair of nodes in movement
26	BER_SPA_sparse_3	Calculate BER with 10 pair of nodes
27	BER_Sensor_net_flooding_4	Calculate BER with 100 pair of nodes
28	BER_EP_optim_sparse_5	Optimize BER with EP technique
29	BER_Zone_2_path_finding_6	(Obsoleted) in area 2, calculate shortest path with 10 pair of nodes
30	BER_GA_optim_zone2_7	(Obsoleted) in area 2, optimize connections with EP

C1: Test_3_City_EP_Dynamic_mode

In this Matlab source file, a step-by-step execution of how to find an optimal route between two randomly selected pair of nodes is demonstrated. It consists of 4 parts: distance matrix initialization, initial fitness calculation, crossover and mutation, new fitness calculation. The output of this file is a plot of 20 iterations of optimization results.

```
%================================================================%

%=================Evolutionary Programming====================%

%================================================================%

%======================Optimization=========================%

%================================================================%

%=========================Step 1===========================%

%================================================================%

%==================Distance   Matrix==========================%

%================================================================%

Zone_triangle_set =zeros(34,4);
```

```
% Define central nodes of each zone

Zone_central_node_set =[2;3;4;5;6;7;8;9;10;13;17;19;20;21;22;23;24;26;27;...

    31;32;35;36;37;38;39;41;43;51;52;53;54;55;56;57];

% Find 3 sensor neighbors for each central node

for i=1:34

    if isempty(find(Left_street_node==Zone_central_node_set(i,1),1,'first'))

        Zone_triangle_set(i,1)=0;

    else


Zone_triangle_set(i,1)=find(Left_street_node==Zone_central_node_set(i,1),1,'fir
st');

    end

    if isempty(find(Right_street_node==Zone_central_node_set(i,1),1,'first'));

        Zone_triangle_set(i,2)=0;

    else


Zone_triangle_set(i,2)=find(Right_street_node==Zone_central_node_set(i,1),1,'f
irst')+39;

    end

    if isempty(find(Upper_street_pair(:,1)==Zone_central_node_set(i,1),1,'first'))
```

```
        Zone_triangle_set(i,3)=0;

    else

Zone_triangle_set(i,3)=find(Upper_street_pair(:,1)==Zone_central_node_set(i,1
),1,'first')+81;

    end

    if isempty(find(Lower_street_pair(:,1)==Zone_central_node_set(i,1),1,'first'))

        Zone_triangle_set(i,4)=0;

    else

Zone_triangle_set(i,4)=find(Lower_street_pair(:,1)==Zone_central_node_set(i,1
),1,'first')+126;

    end

end

Zone_triangle_set(13,:)=[14,144,145,0];

Zone_triangle_set = sort(Zone_triangle_set,2);

for i=1:34

    if Zone_triangle_set(i,1)==0

        Zone_triangle_set(i,1:3)=Zone_triangle_set(i,2:4);

    end

end
```

```
Zone_triangle_set=Zone_triangle_set(:,1:3);

% Calculate Distance Matrix between node

DM_Area_1 = zeros(57,57);

for i=1:56

  for j=i+1:57

    for k=1:length(Area_1_street)

      if Area_1_street(k,1)==i && Area_1_street(k,2)==j

          DM_Area_1(i,j)=(Area_1_node_X_Y(i,1)-...

          Area_1_node_X_Y(j,1))^2+...

          (Area_1_node_X_Y(i,2)-...

          Area_1_node_X_Y(j,2))^2;

          DM_Area_1(i,j)=sqrt(DM_Area_1(i,j));

         DM_Area_1(j,i)=DM_Area_1(i,j);

      end

    end

  end

end

Dist_max = max(max(DM_Area_1));
```

```
gap = Dist_max/8;

% Generate two matrices: BER matrix and Sensor_no matrix

DM_Sensor_BER = zeros(102,102);

DM_Sensor_no = zeros(102,102);

% Inner-triangle DM value assignment

for i =1:34

  % Sensor neighbors have BER = 1E-4

  DM_Sensor_BER ((i-1)*3+1,(i-1)*3+2)=0.0001;

  DM_Sensor_BER ((i-1)*3+1,i*3)=0.0001;

  DM_Sensor_BER ((i-1)*3+2,(i-1)*3+1)=0.0001;

  DM_Sensor_BER ((i-1)*3+2,i*3)=0.0001;

  DM_Sensor_BER (i*3,(i-1)*3+1)=0.0001;

  DM_Sensor_BER (i*3,(i-1)*3+2)=0.0001;

  % Sensor neighbors have Sensor number = 1

  DM_Sensor_no ((i-1)*3+1,(i-1)*3+2)=1;

  DM_Sensor_no ((i-1)*3+1,i*3)=1;

  DM_Sensor_no ((i-1)*3+2,(i-1)*3+1)=1;

  DM_Sensor_no ((i-1)*3+2,i*3)=1;
```

```
  DM_Sensor_no (i*3,(i-1)*3+1)=1;

  DM_Sensor_no (i*3,(i-1)*3+2)=1;

end

% Neighbor triangle assignment

Left_zone_pair = [1:9,15:20,23:33];

for i=1:length(Left_zone_pair)

   Temp_dist = (Area_1_sensor_X_Y(1,Zone_triangle_set(Left_zone_pair(1,i),2))...

      -Area_1_sensor_X_Y(1,Zone_triangle_set(Left_zone_pair(1,i)+1,1)))^2+...

      (Area_1_sensor_X_Y(2,Zone_triangle_set(Left_zone_pair(1,i),2))...

      -Area_1_sensor_X_Y(2,Zone_triangle_set(Left_zone_pair(1,i)+1,1)))^2;

   DM_Sensor_no(3*Left_zone_pair(1,i)+1,3*(Left_zone_pair(1,i)-1)+2)=...

      round(sqrt(Temp_dist)/25)+1;

   DM_Sensor_no(3*(Left_zone_pair(1,i)-1)+2,3*Left_zone_pair(1,i)+1)=...
round(sqrt(Temp_dist)/25)+1;

   Temp_dist = sqrt(Temp_dist)/gap;

   DM_Sensor_BER(3*Left_zone_pair(1,i)+1,3*(Left_zone_pair(1,i)-
1)+2)=Temp_dist*1E-4;

   DM_Sensor_BER(3*(Left_zone_pair(1,i)-
1)+2,3*Left_zone_pair(1,i)+1)=Temp_dist*1E-4;

end

Up_down_zone_pair = [4,15;8,18;9,19;10,11;12,16;13,17;14,21;25,30];
```

```
for i=1:length(Up_down_zone_pair)

  Temp_dist =
(Area_1_sensor_X_Y(1,Zone_triangle_set(Up_down_zone_pair(i,1),3))...

    -Area_1_sensor_X_Y(1,Zone_triangle_set(Up_down_zone_pair(i,2),3)))^2+...

    (Area_1_sensor_X_Y(2,Zone_triangle_set(Up_down_zone_pair(i,1),3))...

    -Area_1_sensor_X_Y(2,Zone_triangle_set(Up_down_zone_pair(i,1),3)))^2;

  DM_Sensor_no(3*Up_down_zone_pair(i,1),3*Up_down_zone_pair(i,2))=...

    round(sqrt(Temp_dist)/25)+1;

  DM_Sensor_no(3*Up_down_zone_pair(i,1),3*Up_down_zone_pair(i,2))=...

    round(sqrt(Temp_dist)/25)+1;

    Temp_dist = sqrt(Temp_dist)/gap;


DM_Sensor_BER(3*Up_down_zone_pair(i,1),3*Up_down_zone_pair(i,2))=Temp
_dist*1E-4;


DM_Sensor_BER(3*Up_down_zone_pair(i,2),3*Up_down_zone_pair(i,1))=Temp
_dist*1E-4;

end

%============================================================%

%=====================Step:   2============================%

%============================================================%

%===================Initial   Fitness==========================%
```

```
%===============================================================%

%===============================================================%

Initial_population_binary = Initial_population_binary_backup;

Initial_BER_fitness = zeros(Population_size,Fitness_number);

Optimized_BER_fitness = zeros(Population_size,Fitness_number);

Initial_Sensor_fitness = zeros(Population_size, Fitness_number);

Optimized_Sensor_fitness = zeros(Population_size, Fitness_number);

for n=1:Fitness_number

  Initial_start_node = Node_1_X_Y_speed_A(:,n);

  Initial_end_node = Node_2_X_Y_speed_A(:,n);

  Initial_population_binary = Initial_population_binary_backup;

  % Locate nearest zone near starting node

  pos = 1;

  Dist = 1E6;

  for i=1:Zone_size

    dist=(Initial_start_node(1,1)-
Area_1_node_X_Y(Zone_central_node_set(i,1),1))^2+...

      (Initial_start_node(2,1)-
Area_1_node_X_Y(Zone_central_node_set(i,1),2))^2;

    if Dist>dist
```

```
        Dist=dist;

        pos=i;

      end

    end

    Initial_start_zone = pos;

    % Locate nearest zone near ending node

    pos = 1;

    Dist = 1E6;

    for i=1:Zone_size

      dist=(Initial_end_node(1,1)-
  Area_1_node_X_Y(Zone_central_node_set(i,1),1))^2+...

        (Initial_end_node(2,1)-
  Area_1_node_X_Y(Zone_central_node_set(i,1),2))^2;

      if Dist>dist

        Dist=dist;

        pos=i;

      end

    end

    Initial_end_zone = pos;

    if Initial_start_zone > Initial_end_zone
```

```
    Temp = Initial_end_zone;

    Initial_end_zone = Initial_start_zone;

    Initial_start_zone = Temp;

end

% Two seperate fitness values are calculated: one is stored in Initial_

% BER_fitness, recording link's bit error rate, another one is stored

% in Initial_Sensor_fitness, recording link's usage of sensors

% Line: 316 to 524

for i=1:Population_size

    pos = Initial_start_zone;

    j = 1;

    while pos ~= Initial_end_zone && j < Zone_size

        % Calculate the fitness:

        % 001--path jumps over zone, bit error rate and sensor no value

        % only related with street among neighboring zones

        if Initial_population_binary(i,(j-1)*3+1:j*3) == logical([0 0 1])

            Initial_BER_fitness(i,n)=Initial_BER_fitness(i,n)+...

                DM_Area_1(Zone_central_node_set(pos,1),...

                Zone_central_node_set(pos+1,1))/gap*1E-4;
```

```
    Initial_Sensor_fitness(i,n)=Initial_Sensor_fitness(i,n)+...
      round(DM_Area_1(Zone_central_node_set(pos,1),...
      Zone_central_node_set(pos+1,1))/25);
        if pos < Zone_size-1
          pos = pos + 1;
        else
          pos = pos -1;
        end
  elseif Initial_population_binary(i,(j-1)*3+1:j*3) == logical([0 1 0])
    sensor_pos = (pos-1)*3+1;
    Temp_list = DM_Sensor_BER(sensor_pos,:);
    Temp_list(1,sensor_pos:sensor_pos+2)=zeros(1,3);
    Temp_list = sort (Temp_list);
    Temp_pos = find(Temp_list>0,1,'first');
    if isempty(Temp_pos)
      % Temp_pos is empty, means current sensor does not have
      % a neighbor for signal transmission, the algorithm
      % will jump to neighboring zone automatically
      Initial_BER_fitness(i,n)=Initial_BER_fitness(i,n)+...
```

```
        DM_Area_1(Zone_central_node_set(pos,1),...
        Zone_central_node_set(pos+1,1))/gap*1E-4;
        Initial_Sensor_fitness(i,n)=Initial_Sensor_fitness(i,n)+...
        round(DM_Area_1(Zone_central_node_set(pos,1),...
        Zone_central_node_set(pos+1,1))/25);
          if pos < Zone_size - 1
            pos = pos + 1;
          else
            pos = pos -1;
          end
      else
        zone_pos = find(DM_Sensor_BER(sensor_pos,:)==Temp_list(1,Temp_pos),1,'first');
        pos = ceil(zone_pos/3);
        Initial_BER_fitness(i,n)=Initial_BER_fitness(i,n)+...
        DM_Sensor_BER(sensor_pos,zone_pos)+1E-4;
        Initial_Sensor_fitness(i,n)=Initial_Sensor_fitness(i,n)+...
        1+DM_Sensor_no(sensor_pos,zone_pos);
      end
    elseif Initial_population_binary(i,(j-1)*3+1:j*3) == logical([0 1 1])
```

```
sensor_pos = (pos-1)*3+2;

Temp_list = DM_Sensor_BER(sensor_pos,:);

Temp_list(1,sensor_pos-1:sensor_pos+1)=zeros(1,3);

Temp_list = sort (Temp_list) ;

Temp_pos = find(Temp_list>0,1,'first');

if isempty(Temp_pos)

   Initial_BER_fitness(i,n)=Initial_BER_fitness(i,n)+...

   DM_Area_1(Zone_central_node_set(pos,1),...

   Zone_central_node_set(pos+1,1))/gap*1E-4;

   Initial_Sensor_fitness(i,n)=Initial_Sensor_fitness(i,n)+...

   round(DM_Area_1(Zone_central_node_set(pos,1),...

   Zone_central_node_set(pos+1,1))/25);

      if pos < Zone_size - 1

         pos = pos + 1;

      else

         pos = pos -1;

      end

else

   zone_pos                                                    =
find(DM_Sensor_BER(sensor_pos,:)==Temp_list(1,Temp_pos),1,'first');
```

```
            pos = ceil(zone_pos/3);
            Initial_BER_fitness(i,n)=Initial_BER_fitness(i,n)+...
            DM_Sensor_BER(sensor_pos,zone_pos)+1E-4;
            Initial_Sensor_fitness(i,n)=Initial_Sensor_fitness(i,n)+...
            1+DM_Sensor_no(sensor_pos,zone_pos);
        end
    elseif Initial_population_binary(i,(j-1)*3+1:j*3) == logical([1 0 0])
        sensor_pos = (pos-1)*3+1;
        Temp_list = DM_Sensor_BER(sensor_pos,:);
        Temp_list(1,sensor_pos:sensor_pos+2)=zeros(1,3);

        Temp_list = sort (Temp_list);
        Temp_pos = find(Temp_list>0,1,'first');
        if isempty(Temp_pos)
            Initial_BER_fitness(i,n)=Initial_BER_fitness(i,n)+...
            DM_Area_1(Zone_central_node_set(pos,1),...
            Zone_central_node_set(pos+1,1))/gap*1E-4;
            Initial_Sensor_fitness(i,n)=Initial_Sensor_fitness(i,n)+...
            round(DM_Area_1(Zone_central_node_set(pos,1),...
```

```
Zone_central_node_set(pos+1,1))/25);

if pos < Zone_size - 1

pos = pos + 1;

else

pos = pos -1;

end

else

zone_pos =
find(DM_Sensor_BER(sensor_pos,:)==Temp_list(1,Temp_pos),1,'first');

pos = ceil(zone_pos/3);

Initial_BER_fitness(i,n)=Initial_BER_fitness(i,n)+...

DM_Sensor_BER(sensor_pos,zone_pos)+2E-4;

Initial_Sensor_fitness(i,n)=Initial_Sensor_fitness(i,n)+...

2+DM_Sensor_no(sensor_pos,zone_pos);

end

else

sensor_pos = (pos-1)*3+2;

Temp_list = DM_Sensor_BER(sensor_pos,:);

Temp_list(1,sensor_pos-1:sensor_pos+1)=zeros(1,3);

Temp_list = sort (Temp_list) ;
```

```
Temp_pos = find(Temp_list>0,1,'first');

if isempty(Temp_pos)

  Initial_BER_fitness(i,n)=Initial_BER_fitness(i,n)+...

  DM_Area_1(Zone_central_node_set(pos,1),...

  Zone_central_node_set(pos+1,1))/gap*1E-4;

  Initial_Sensor_fitness(i,n)=Initial_Sensor_fitness(i,n)+...

  round(DM_Area_1(Zone_central_node_set(pos,1),...

  Zone_central_node_set(pos+1,1))/25);

    if pos < Zone_size - 1

      pos = pos + 1;

    else

      pos = pos -1;

    end

else

  zone_pos                                                    =
find(DM_Sensor_BER(sensor_pos,:)==Temp_list(1,Temp_pos),1,'first');

  pos = ceil(zone_pos/3);

  Initial_BER_fitness(i,n)=Initial_BER_fitness(i,n)+...

  DM_Sensor_BER(sensor_pos,zone_pos)+2E-4;

  Initial_Sensor_fitness(i,n)=Initial_Sensor_fitness(i,n)+...
```

```
        2+DM_Sensor_no(sensor_pos,zone_pos);
      end
    end
    j = j + 1;
  end
  % Calculate BER at transmitter side
  Dist_1 = (Initial_start_node(1,1)-...
    Area_1_node_X_Y(Zone_central_node_set(Initial_start_zone,1),1))^2....
      +(Initial_start_node(2,1)-...
    Area_1_node_X_Y(Zone_central_node_set(Initial_start_zone,1),2))^2;
  % Calculate BER at receiver side
  Dist_2 = (Initial_end_node(1,1)-...
    Area_1_node_X_Y(Zone_central_node_set(Initial_end_zone,1),1))^2....
      +(Initial_end_node(2,1)-...
    Area_1_node_X_Y(Zone_central_node_set(Initial_end_zone,1),2))^2;
  % Calculate the cummulative BER rate
  Initial_BER_fitness(i,n)=Initial_BER_fitness(i,n)+...
round(sqrt(Dist_1+Dist_2)/gap)*1E-4;
  Initial_Sensor_fitness(i,n)=Initial_Sensor_fitness(i,n)+...
```

```
            round(sqrt(Dist_1+Dist_2)/25);

  end

 %==========================================================%

%========================Step:  3===========================%

%==========================================================%

%==========Single  objective  optimization========================%

%==========================================================%

%==================Target:  BER=============================%

%==========================================================%

%==========================================================%

%==========================================================%

%==========================================================%

   Evolution_round = 20;

  BER_fitness_table = zeros(Population_size,Evolution_round);

  Sensor_fitness_table = zeros(Population_size,Evolution_round);

  % Assign initial fitness value to fitness table column 1

  BER_fitness_table(:,1)=Initial_BER_fitness(:,n);

  Initial_BER_fitness_1 = Initial_BER_fitness(:,n);

  Sensor_fitness_table(:,1)=Initial_Sensor_fitness(:,n);
```

```
for m = 2: Evolution_round

%===========================================================%

%============Survival probability calculation====================%

%===========================================================%

   Survival_probability = Initial_BER_fitness_1;

   Survival_probability = Survival_probability ./ sum(Survival_probability);

%===========================================================%

%===Evolutional programming: keep half best into next generation======%

%===========================================================%

%============Not random roulette wheel spin!!======= ============%

%===========================================================%

%===========================================================%

    New_population_binary=logical(zeros(Population_size, 3*Zone_size));

% Copy 5 best populations from initial population to new population

   for i = 1:5

       pos = find(Survival_probability == max(Survival_probability),1,'first');

       New_population_binary(i,:) = Initial_population_binary(pos,:);

       Survival_probability(pos,1) = 0;

   end
```

```
%==========================================================%

%===========Copy the others into another pool==================%

%==========================================================%

%==========Then crossover with best populations================%

%==========================================================%

%==========================================================%

Crossover_population_binary                    =logical(zeros(Population_size,
3*Zone_size));

% Copy 5 bad populations from initial population to new population

for i = 1:5

  Crossover_population_binary(2*i-1,:) = New_population_binary(i,:);

  pos = find(Survival_probability > 0,1,'first');

  Crossover_population_binary(2*i,:) = Initial_population_binary(pos,:);

  Survival_probability(pos,1) = 0;

end

% Generate a random vector to define crossover positions

Crossover_pos = randi([1,Zone_size-1],1,Population_size/2);

% Perform crossover operation

for i = 1:5

  %================First part crossover==================%
```

```
        Temp_vector=Crossover_population_binary(2*i-
1,1:3*Crossover_pos(1,i));

        Crossover_population_binary(2*i-1,1:3*Crossover_pos(1,i))...

          = Crossover_population_binary(2*i,1:3*Crossover_pos(1,i));

      Crossover_population_binary(2*i,1:3*Crossover_pos(1,i)) = Temp_vector;

        %================Second part crossover===================%

        Temp_vector=Crossover_population_binary(2*i-
1,3*Crossover_pos(1,i)+1:3*Zone_size);

        Crossover_population_binary(2*i-
1,3*Crossover_pos(1,i)+1:3*Zone_size)...

          =
Crossover_population_binary(2*i,3*Crossover_pos(1,i)+1:3*Zone_size);

        Crossover_population_binary(2*i,3*Crossover_pos(1,i)+1:3*Zone_size) =
Temp_vector;

    end

  % Perform mutation operation

    Mutation_probability = 0.01;

    % Be careful here: mutation is over zones, instead of bits!!!

    % The total number of zones is Population_size * Zone_size, thus we

    % generate a vector with many random numbers

     Mutation_values = rand(Population_size, Zone_size);

    % Find all zones whose mutation_values are less than 0.01
```

```
[Mutation_pos_row, Mutation_pos_col] = find(Mutation_values < Mutation_probability);

  if isempty(Mutation_pos_row)==0

   for i = 1: length(Mutation_pos_row)

     % Zone mutation operation!!

      % Random number from [1, 5] then digitize into 3 bits

       pos = randi([1,5]);

Crossover_population_binary(Mutation_pos_row(i,1),3*Mutation_pos_col(i,1):...

         -1:3*Mutation_pos_col(i,1)-2) = dec2binvec(pos,3);

     end

  end

  %=========================================================%

  %================New  fitness  calculation=======================%

  %=========================================================%

  %====================Target:  BER============================%

  %=========================================================%

  %=========================================================%

  New_BER_fitness = zeros(Population_size,1);

  New_Sensor_fitness = zeros(Population_size,1);
```

```
for i=1:Population_size
    pos = Initial_start_zone;
    j = 1;
    while pos ~= Initial_end_zone && j < Zone_size
      % Calculate the fitness:
      % 001--path jumps over zone, bit error rate and sensor no value
      % only related with street among neighboring zones
      if Crossover_population_binary(i,(j-1)*3+1:j*3) == logical([0 0 1])
          New_BER_fitness(i,1)=New_BER_fitness(i,1)+...
            DM_Area_1(Zone_central_node_set(pos,1),...
            Zone_central_node_set(pos+1,1))/gap*1E-4;
         New_Sensor_fitness(i,1)=New_Sensor_fitness(i,1)+...
         round(DM_Area_1(Zone_central_node_set(pos,1),...
         Zone_central_node_set(pos+1,1))/25);
            if pos < Zone_size - 1
               pos = pos + 1;
            else
               pos = pos -1;
```

```
        end

    elseif Crossover_population_binary(i,(j-1)*3+1:j*3) == logical([0 1 0])

      sensor_pos = (pos-1)*3+1;

      Temp_list = DM_Sensor_BER(sensor_pos,:);

      Temp_list(1,sensor_pos:sensor_pos+2)=zeros(1,3);

       Temp_list = sort (Temp_list) ;

       Temp_pos = find(Temp_list>0,1,'first');

       if isempty(Temp_pos)

          % Temp_pos is empty, means current sensor does not have

         % a neighbor for signal transmission, the algorithm

         % will jump to neighboring zone automatically

          New_BER_fitness(i,1)=New_BER_fitness(i,1)+...

         DM_Area_1(Zone_central_node_set(pos,1),...

         Zone_central_node_set(pos+1,1))/gap*1E-4;

         New_Sensor_fitness(i,1)=New_Sensor_fitness(i,1)+...

         round(DM_Area_1(Zone_central_node_set(pos,1),...

         Zone_central_node_set(pos+1,1))/25);

         if pos < Zone_size - 1
```

```
            pos = pos + 1;
          else
            pos = pos -1;
          end
        else
          zone_pos                                                    =
find(DM_Sensor_BER(sensor_pos,:)==Temp_list(1,Temp_pos),1,'first');
          pos = ceil(zone_pos/3);
          New_BER_fitness(i,1)=New_BER_fitness(i,1)+...
          DM_Sensor_BER(sensor_pos,zone_pos)+1E-4;
          New_Sensor_fitness(i,1)=New_Sensor_fitness(i,1)+...
          1+DM_Sensor_no(sensor_pos,zone_pos);
        end
      elseif Crossover_population_binary(i,(j-1)*3+1:j*3) == logical([0 1 1])
        sensor_pos = (pos-1)*3+2;
        Temp_list = DM_Sensor_BER(sensor_pos,:);
        Temp_list(1,sensor_pos-1:sensor_pos+1)=zeros(1,3);
        Temp_list = sort (Temp_list) ;
        Temp_pos = find(Temp_list>0,1,'first');
```

```
if isempty(Temp_pos)
     New_BER_fitness(i,1)=New_BER_fitness(i,1)+...
    DM_Area_1(Zone_central_node_set(pos,1),...
    Zone_central_node_set(pos+1,1))/gap*1E-4;
    New_Sensor_fitness(i,1)=New_Sensor_fitness(i,1)+...
    round(DM_Area_1(Zone_central_node_set(pos,1),...
    Zone_central_node_set(pos+1,1))/25);
    if pos < Zone_size - 1
        pos = pos + 1;
    else
        pos = pos -1;
    end
else
    zone_pos                                                        =
find(DM_Sensor_BER(sensor_pos,:)==Temp_list(1,Temp_pos),1,'first');
    pos = ceil(zone_pos/3);
     New_BER_fitness(i,1)=New_BER_fitness(i,1)+...
    DM_Sensor_BER(sensor_pos,zone_pos)+1E-4;
    New_Sensor_fitness(i,1)=New_Sensor_fitness(i,1)+...
    1+DM_Sensor_no(sensor_pos,zone_pos);
```

```
    end

  elseif Crossover_population_binary(i,(j-1)*3+1:j*3) == logical([1 0 0])

     sensor_pos = (pos-1)*3+1;

     Temp_list = DM_Sensor_BER(sensor_pos,:);

     Temp_list(1,sensor_pos:sensor_pos+2)=zeros(1,3);

     Temp_list = sort (Temp_list) ;

     Temp_pos = find(Temp_list>0,1,'first');

     if isempty(Temp_pos)

        New_BER_fitness(i,1)=New_BER_fitness(i,1)+...

       DM_Area_1(Zone_central_node_set(pos,1),...

       Zone_central_node_set(pos+1,1))/gap*1E-4;

       New_Sensor_fitness(i,1)=New_Sensor_fitness(i,1)+...

       round(DM_Area_1(Zone_central_node_set(pos,1),...

       Zone_central_node_set(pos+1,1))/25);

       if pos < Zone_size - 1

          pos = pos + 1;

       else

          pos = pos -1;
```

```
        end
      else
        zone_pos                                                    =
find(DM_Sensor_BER(sensor_pos,:)==Temp_list(1,Temp_pos),1,'first');
        pos = ceil(zone_pos/3);
        New_BER_fitness(i,1)=New_BER_fitness(i,1)+...
        DM_Sensor_BER(sensor_pos,zone_pos)+2E-4;
        New_Sensor_fitness(i,1)=New_Sensor_fitness(i,1)+...
        2+DM_Sensor_no(sensor_pos,zone_pos);
      end
    else
      sensor_pos = (pos-1)*3+2;
      Temp_list = DM_Sensor_BER(sensor_pos,:);
      Temp_list(1,sensor_pos-1:sensor_pos+1)=zeros(1,3);
      Temp_list = sort (Temp_list) ;
      Temp_pos = find(Temp_list>0,1,'first');
      if isempty(Temp_pos)
        New_BER_fitness(i,1)=New_BER_fitness(i,1)+...
        DM_Area_1(Zone_central_node_set(pos,1),...
        Zone_central_node_set(pos+1,1))/gap*1E-4;
```

```
New_Sensor_fitness(i,1)=New_Sensor_fitness(i,1)+...
round(DM_Area_1(Zone_central_node_set(pos,1),...
Zone_central_node_set(pos+1,1))/25);
if pos < Zone_size - 1
    pos = pos + 1;
else
    pos = pos -1;
end
else
zone_pos = find(DM_Sensor_BER(sensor_pos,:)==Temp_list(1,Temp_pos),1,'first');
pos = ceil(zone_pos/3);
New_BER_fitness(i,1)=New_BER_fitness(i,1)+...
DM_Sensor_BER(sensor_pos,zone_pos)+2E-4;
New_Sensor_fitness(i,1)=New_Sensor_fitness(i,1)+...
2+DM_Sensor_no(sensor_pos,zone_pos);
end
end
```

```
    j = j + 1;
  end
   % Calculate BER at transmitter side
  Dist_1 = (Initial_start_node(1,1)-...
    Area_1_node_X_Y(Zone_central_node_set(Initial_start_zone,1),1))^2....
      +(Initial_start_node(2,1)-...
    Area_1_node_X_Y(Zone_central_node_set(Initial_start_zone,1),2))^2;
  % Calculate BER at receiver side
  Dist_2 = (Initial_end_node(1,1)-...
    Area_1_node_X_Y(Zone_central_node_set(Initial_end_zone,1),1))^2....
      +(Initial_end_node(2,1)-...
    Area_1_node_X_Y(Zone_central_node_set(Initial_end_zone,1),2))^2;
   % Calculate the cummulative BER rate
   New_BER_fitness(i,1)=New_BER_fitness(i,1)+...
        round(sqrt(Dist_1+Dist_2)/gap)*1E-4;
  New_Sensor_fitness(i,1)=New_Sensor_fitness(i,1)+...
        round(sqrt(Dist_1+Dist_2)/25);
end
```

```
%=============================================================%
%===============Update Fitness Table========================%
%=============================================================%
%=================Select best populations====================%
%=============================================================%
%=============================================================%
 Sorted_fitness_value = zeros(2*Population_size,1);
Sensor_fitness_value = zeros(2*Population_size,1);
Sorted_fitness_value(1:Population_size,1)=Initial_BER_fitness_1;
Sensor_fitness_value(1:Population_size,1)=Initial_Sensor_fitness(:,n);
 Sorted_fitness_value(Population_size+1:...
    2*Population_size,1)=New_BER_fitness;
Sensor_fitness_value(Population_size+1:...
    2*Population_size,1)=New_Sensor_fitness;
 Sorted_fitness_value = sort(Sorted_fitness_value);
for i = 1:10
    BER_fitness_table(i,m)=Sorted_fitness_value(i,1);
    pos = find(Initial_BER_fitness_1 == Sorted_fitness_value(i,1),1,'first');
```

```
   if isempty(pos)
      pos = find(New_BER_fitness == Sorted_fitness_value(i,1),1,'first');
      New_population_binary(i,:) = Crossover_population_binary(pos,:);
      Sensor_fitness_table(i,m)=Sensor_fitness_value(Population_size+pos,1);
   else
       New_population_binary(i,:) = Initial_population_binary(pos,:);
      Sensor_fitness_table(i,m)=Sensor_fitness_value(pos,1);
   end
 end
 %==========================================================%
%==================Update  populations======================%
%==========================================================%
%==========Replace initial population with new population==========%
%==========================================================%
%==========================================================%
 Initial_BER_fitness_1 = Sorted_fitness_value(1:Population_size,1);
Initial_population_binary = New_population_binary;
end
```

```
   Optimized_BER_fitness(:,n)=BER_fitness_table(:,Evolution_round);
      Optimized_Sensor_fitness(:,n)=Sensor_fitness_table(:,Evolution_round);
 end
% Plot BER curves
% X coordinate: number of evolutions
% Y coordinate: average BER value
x = 1: Fitness_number;
y1 = zeros(Fitness_number,1);
y2 = zeros(Fitness_number,1);
for i = 1:Fitness_number
   y1(i,1)=sum(Initial_BER_fitness(:,i))/10;
   y2(i,1)=sum(Optimized_BER_fitness(:,i))/10;
end
plot(x,y1,'--bs','LineWidth',2,...
             'MarkerEdgeColor','k',...
             'MarkerFaceColor','y',...
             'MarkerSize',10);
hold on
```

```
plot(x,y2,'-.ok','LineWidth',2,...
              'MarkerEdgeColor','k',...
              'MarkerFaceColor','r',...
              'MarkerSize',10);
pause;
clf;
 for i = 1:Fitness_number
      y1(i,1)=sum(Initial_Sensor_fitness(:,i))/10;
       y2(i,1)=sum(Optimized_Sensor_fitness(:,i))/10;
   end
 plot(x,y1,'--bs','LineWidth',2,...
              'MarkerEdgeColor','k',...
              'MarkerFaceColor','y',...
              'MarkerSize',10);
hold on
plot(x,y2,'-.ok','LineWidth',2,...
              'MarkerEdgeColor','k',...
              'MarkerFaceColor','r',...
              'MarkerSize',10);
```

pause;

clf;

C.2 Data Loss Rate Analysis

In this MATLAB source file, equation 5.1 is applied to calculate data loss rate of a certain area with Poisson density λ. The input to this function is λ and the output is a plot whose x-axis is hop number, and y-axis is data loss rate.

```
% This function uses a mathematical model to calculate

% Each subarea's data loss rate

% According to its density factor lamda

% Input: Lamda_subarea

% Output: Curves with x-axis: hops, and y-axis: data loss rate

No_subarea = 7;

Lamda_subarea = [0.5278,0.4776,0.3793,0.5584,0.9412,0.8,0.32];

T = 0.1;

Pe = 0:0.01:0.1;

 DSR_subareas = zeros(No_subarea,10);

for i = 1:No_subarea

   for j = 1:10

      DSR_subareas(i,j)=1/(1+sqrt(T/log(1/(1-Pe(1,j))))*atan(sqrt(T/log(1/(1-Pe(1,j))))));

      DSR_subareas(i,j) = DSR_subareas(i,j) * exp(Lamda_subarea(1,i)*(Pe(1,j)-1));
```

```
        end
    plot(1:10, DSR_subareas(i,:));
    saveas(gca,['Data_loss_rate_subarea_',int2str(i)],'tiff');
    pause;
end
```