ENERGY CONSERVATION FOR WIRELESS AD HOC ROUTING

by

Xiaobing Hou

M.Engr. in Signal & Information Processing, 1998

B.Engr. in Information Engineering, 1994

Both from Northern Jiaotong University, Beijing, China

Submitted to the Graduate Faculty of

the Department of Information Science & Telecommunications in

partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

University of Pittsburgh

2006

UNIVERSITY OF PITTSBURGH

DEPARTMENT OF INFORMATION SCIENCE & TELECOMMUNICATIONS

This dissertation was presented

by

Xiaobing Hou

It was defended on

April 5th 2006

and approved by

Dr. David Tipper, Dept. of Information Science & Telecommunications

Dr. Richard Thompson, Dept. of Information Science & Telecommunications

Dr. Joseph Kabara, Dept. of Information Science & Telecommunications

Dr. Daniel Mosse, Dept. of Computer Science

Dr. A. Bruce McDonald, Northeastern University

Dissertation Director: Dr. David Tipper, Dept. of Information Science & Telecommunications

ENERGY CONSERVATION FOR WIRELESS AD HOC ROUTING

Xiaobing Hou, PhD

University of Pittsburgh, 2006

Self-configuring wireless ad hoc networks have attracted considerable attention in the last few years due to their valuable civil and military applications. One aspect of such networks that has been studied insufficiently is the energy efficiency. Energy efficiency is crucial to prolong the network lifetime and thus make the network more survivable.

Nodes in wireless ad hoc networks are most likely to be driven by battery and hence operate on an extremely frugal energy budget. Conventional ad hoc routing protocols are focused on handling the mobility instead of energy efficiency. Energy efficient routing strategies proposed in literature either do not take advantage of sleep modes to conserve energy more efficiently, or incur much overhead in terms of control message and computing complexity to schedule sleep modes and thus are not scalable.

In this dissertation, a novel strategy is proposed to manage the sleep of the nodes in the network so that energy can be conserved and network connectivity can be kept. The novelty of the strategy is its extreme simplicity. The idea is derived from the results of the percolation theory, typically called *gossiping*. Gossiping is a convenient and effective approach and has been successfully applied to several areas of the networking. In the proposed work, we will develop a sleep management protocol from gossiping for both static and mobile wireless ad hoc networks. Then the protocol will be extended to the asynchronous network, where nodes manage their own states independently. Analysis and simulations will be conducted to show the correctness, effectiveness and efficiency of the proposed work. The comparison between analytical and simulation results will justify them for each other. We will investigate the most important performance aspects concerning the proposed strategy, including the effect of parameter tuning and the impacts of routing

protocols. Furthermore, multiple extensions will be developed to improve the performance and make the proposed strategy apply to different network scenarios.

TABLE OF CONTENTS

1.0 INTRODUCTION 1

1.1 Ad Hoc Network Overview 1

1.2 Ad Hoc Network Routing 2

1.3 Sensor Network Routing 3

1.4 Energy Conservation 5

1.5 The Problem Statement 7

1.6 Organization 7

2.0 LITERATURE REVIEW 9

2.1 Ad Hoc Network Routing 9

2.1.1 Proactive Routing 9

2.1.2 Reactive Routing 11

2.1.3 Hybrid Routing 13

2.1.4 Cluster-based Routing 15

2.2 Sensor Network Routing 17

2.2.1 Flooding 18

2.2.2 Forwarding 18

2.2.3 Data-Centric 19

2.3 Energy Efficient Routing 21

2.3.1 Metric-based Routing 21

2.3.2 Sleep-based Routing 23

2.3.3 Discussion 27

2.4 Summary 29

3.0 GOSSIP-BASED SLEEP PROTOCOL 31
3.1 Background on Percolation Theory 31
3.2 Gossip-based Ad Hoc Routing 33
3.3 Gossip-based Sleep Protocol (GSP) 34
3.4 Connectivity Study 38
3.4.1 Simulation Model 38
3.4.2 Results 39
3.5 Performance Analysis 41
3.5.1 Radio Model 41
3.5.2 GSP Performance Analysis 42
3.5.3 Simulation Model for α 44
3.5.4 Simulation Results for α 46
3.5.5 Continued Performance Analysis 46
3.5.6 Analysis at Frame Level 47
3.6 Summary 48
4.0 PERFORMANCE EVALUATION 50
4.1 Preliminary Simulation 50
4.1.1 Simulation Model 51
4.1.2 Simulation Results 54
4.1.3 Comparison with Analytical Results 58
4.2 Comprehensive Simulation 66
4.2.1 Parameters 66
4.2.2 Simulation Results 66
4.2.2.1 Packet Delivery Fraction 66
4.2.2.2 End-to-end Delay 75
4.3 Discussion 78
4.4 Impacts of Routing Protocols 79
4.4.1 Proactive Protocols 79
4.4.2 Reactive Protocols 80
4.5 Summary 82

5.0 EXTENSIONS OF GSP 83
5.1 Adaptive GSP (A-GSP) 83
5.2 Battery-aware GSP (B-GSP) 87
5.3 Traffic-aware GSP (T-GSP) 92
5.4 Summary 105
6.0 CONCLUSIONS 107
6.1 Contributions 107
6.2 Future Research 108
APPENDIX. PUBLICATIONS FROM THIS WORK 110
BIBLIOGRAPHY 111

LIST OF TABLES

1 Lucent IEEE 802.11 WaveLAN PC Card Characteristics 24

2 Connectivity Measurement of a Graph with Random Mobility 40

3 Radio Characteristics 42

4 Energy Consumption Model for Lucent IEEE 802.11 WaveLAN PC Card with $2Mbps$ 51

5 GSP Simulation Parameters 52

6 Average Path Length with 90% c.i. of GSP1 in 50-node Networks with Sleep Probability=0.3 59

7 Average Path Length with 90% c.i. of GSP1 in 100-node Networks with Sleep Probability=0.5 59

8 Parameter Values Used in the Analytical Model 64

LIST OF FIGURES

1 Power consumption by each subsystem of a Toshiba 410 CDT mobile computer (Pentium 90 with 8MBytes of EDO RAM and AT&T WaveLAN PC Card) 4

2 Percolation probability 32

3 The left-hand curve is a sketch of the mean cluster size $\chi(p)$. The right-hand curve is a sketch of the mean size $\chi^f(p)$ of a finite open cluster when $p > p_c$. 32

4 Nodes switch on and off in GSP 36

5 The central area of the grid topology used by the simulation. 40

6 Radio model. 41

7 The ratio of average extra path length (α) and the nodes disconnected from the sink when p=0.25, with a 90% c.i. 44

8 Energy difference (E_{diff}) between $E_{GSP-saved}$ and $E_{GSP-extra}$ vs. traffic load (B) in bits when $p = 0.25$, $N = 100$, $L_{min} = 5.05$ and $\alpha = 0.156$. 45

9 Number of nodes N vs. traffic load B (bits) when $p = 0.25$, $\alpha = 0.22$ and $E_{diff} = 0$. 45

10 Network lifetime of DSR and GSP1 with different gossip sleep probability (p) . . . 53

11 Average packet delivery fraction of DSR and GSP1 with different gossip sleep probability (p) 53

12 Average end-to-end delay of DSR and GSP1 with different gossip sleep probability (p) 54

13 Network lifetime of DSR, GSP1 and GSP2 with different gossip sleep probability (p) in the network of 100 transit nodes with pause time 0 56

14 Average packet delivery fraction of DSR, GSP1 and GSP2 with different gossip sleep probability (p) in the network of 100 transit nodes with pause time 0 56

15 Network lifetime of DSR and GSP1 awith different gossip period in the network of 100 transit nodes with pause time 0, $p = 0.3$ 57
16 Average packet delivery fraction of DSR and GSP1 with different gossip period in the network of 100 transit nodes with pause time 0, $p = 0.3$ 57
17 Snapshot of DSR at time 161 61
18 Snapshot of GSP at time 161 61
19 Snapshot of DSR at time 260 61
20 Snapshot of GSP at time 260 62
21 Snapshot of DSR at time 300 62
22 Snapshot of GSP at time 300 63
23 Network lifetime comparison between simulation and analytical results in a network of 50 transit nodes with gossip sleep probability p = 0.3 64
24 Network lifetime comparison between simulation and analytical results in a network of 100 transit nodes with gossip sleep probability p = 0.5 65
25 Network lifetime comparison between simulation (packet delivery fraction) and analytical results in a network of 100 transit nodes with gossip sleep probability p = 0.5 65
26 Average packet delivery fraction of DSR (traffic load=5pkt/sec/flow) in the network of 100 transit nodes with different mobility level (pause time) 67
27 Average packet delivery fraction of DSR (traffic load=10pkt/sec/flow) in the network of 100 transit nodes with different mobility level (pause time) 67
28 Average packet delivery fraction of DSR (traffic load=15pkt/sec/flow) in the network of 100 transit nodes with different mobility level (pause time) 68
29 Average packet delivery fraction of DSR (traffic load=20pkt/sec/flow) in the network of 100 transit nodes with different mobility level (pause time) 68
30 Average packet delivery fraction of GSP2 (gossip sleep probability $p = 0.5$, traffic load=5pkt/sec/flow) in the network of 100 transit nodes with different mobility level (pause time) 69

31 Average packet delivery fraction of GSP2 (gossip sleep probability $p = 0.5$, traffic load=10pkt/sec/flow) in the network of 100 transit nodes with different mobility level (pause time) 69
32 Average packet delivery fraction of GSP2 (gossip sleep probability $p = 0.5$, traffic load=15pkt/sec/flow) in the network of 100 transit nodes with different mobility level (pause time) 70
33 Average packet delivery fraction of GSP2 (gossip sleep probability $p = 0.5$, traffic load=20pkt/sec/flow) in the network of 100 transit nodes with different mobility level (pause time) 70
34 Average packet delivery fraction of GSP2 (gossip sleep probability $p = 0.6$, traffic load=5pkt/sec/flow) in the network of 100 transit nodes with different mobility level (pause time) 71
35 Average packet delivery fraction of GSP2 (gossip sleep probability $p = 0.6$, traffic load=10pkt/sec/flow) in the network of 100 transit nodes with different mobility level (pause time) 71
36 Average packet delivery fraction of GSP2 (gossip sleep probability $p = 0.6$, traffic load=15pkt/sec/flow) in the network of 100 transit nodes with different mobility level (pause time) 72
37 Average packet delivery fraction of GSP2 (gossip sleep probability $p = 0.6$, traffic load=20pkt/sec/flow) in the network of 100 transit nodes with different mobility level (pause time) 72
38 Average packet delivery fraction of GSP2 (gossip sleep probability $p = 0.7$, traffic load=5pkt/sec/flow) in the network of 100 transit nodes with different mobility level (pause time) 73
39 Average packet delivery fraction of GSP2 (gossip sleep probability $p = 0.7$, traffic load=10pkt/sec/flow) in the network of 100 transit nodes with different mobility level (pause time) 73
40 Average packet delivery fraction of GSP2 (gossip sleep probability $p = 0.7$, traffic load=15pkt/sec/flow) in the network of 100 transit nodes with different mobility level (pause time) 74

41 Average packet delivery fraction of GSP2 (gossip sleep probability $p = 0.7$, traffic load=20pkt/sec/flow) in the network of 100 transit nodes with different mobility level (pause time) 74
42 Average end-to-end delay of DSR with various traffic load versus nodal pause time in the network of 100 transit nodes 75
43 Zoom of Fig. 42 76
44 Average end-to-end delay of GSP2 (gossip sleep probability $p = 0.5$) with various traffic load versus nodal pause time in the network of 100 transit nodes 76
45 Average end-to-end delay of GSP2 (gossip sleep probability $p = 0.6$) with various traffic load versus nodal pause time in the network of 100 transit nodes 77
46 Average end-to-end delay of GSP2 (gossip sleep probability $p = 0.7$) with various traffic load versus nodal pause time in the network of 100 transit nodes 77
47 A sketch of P_{nb}^{A} 85
48 Examples of P_{nb}^{A} 85
49 An example of random topology 86
50 Network lifetime of AODV, A-GSP and GSP with different gossip sleep probability (p) in a uniformly distributed mobile network of 100 transit nodes, pause time=0 . . 87
51 Average packet delivery fraction of AODV, A-GSP and GSP with different gossip sleep probability (p) in a uniformly distributed mobile network of 100 transit nodes, pause time=0 88
52 Network lifetime of AODV, A-GSP and GSP with different gossip sleep probability (p) in a non-uniformly distributed mobile network of 100 transit nodes, pause time=0 88
53 Average packet delivery fraction of AODV, A-GSP and GSP with different gossip sleep probability (p) in a non-uniformly distributed mobile network of 100 transit nodes, pause time=0 89
54 Average packet delivery fraction of AODV, GSP with $p = 0.7$ and B-GSP with $\bar{p} = 0.7$ in a network of 100 transit nodes with different initial energy, pause time is 0 91
55 Network lifetime of AODV, GSP with $p = 0.7$ and B-GSP with $\bar{p} = 0.7$ in a network of 100 transit nodes with different initial energy, pause time is 0 91

56 Average packet delivery fraction of GSP and T-GSP with long-lived CBR traffic (pause time=0, traffic rate=5pkt/sec) . . . 93
57 Average packet delivery fraction of GSP and T-GSP with long-lived CBR traffic (pause time=0, traffic rate=10pkt/sec) . . . 94
58 Average packet delivery fraction of GSP and T-GSP with long-lived CBR traffic (static network, traffic rate=5pkt/sec) . . . 94
59 Average packet delivery fraction of GSP and T-GSP with long-lived CBR traffic (static network, traffic rate=10pkt/sec) . . . 95
60 Average end-to-end delay of GSP and T-GSP with long-lived CBR traffic (pause time=0, traffic rate=5pkt/sec) . . . 95
61 Average end-to-end delay of GSP and T-GSP with long-lived CBR traffic (pause time=0, traffic rate=10pkt/sec) . . . 96
62 Average end-to-end delay of GSP and T-GSP with long-lived CBR traffic (static network, traffic rate=5pkt/sec) . . . 96
63 Average end-to-end delay of GSP and T-GSP with long-lived CBR traffic (static network, traffic rate=10pkt/sec) . . . 97
64 Network lifetime of GSP and T-GSP with different long-lived CBR traffic (pause time=0) . . . 97
65 Network lifetime of GSP and T-GSP with different long-lived CBR traffic (static network) . . . 98
66 Average packet delivery fraction of GSP and T-GSP with exponential on-off traffic (pause time=0, traffic rate=10pkt/sec) . . . 99
67 Average packet delivery fraction of GSP and T-GSP with exponential on-off traffic (pause time=0, traffic rate=20pkt/sec) . . . 99
68 Average packet delivery fraction of GSP and T-GSP with exponential on-off traffic (static network, traffic rate=10pkt/sec) . . . 100
69 Average packet delivery fraction of GSP and T-GSP with exponential on-off traffic (static network, traffic rate=20pkt/sec) . . . 100
70 Average end-to-end delay of GSP and T-GSP with exponential on-off traffic (pause time=0, traffic rate=10pkt/sec) . . . 101

71 Average end-to-end delay of GSP and T-GSP with exponential on-off traffic (pause time=0, traffic rate=20pkt/sec) . . . 101
72 Average end-to-end delay of GSP and T-GSP with exponential on-off traffic (static network, traffic rate=10pkt/sec) . . . 102
73 Average end-to-end delay of GSP and T-GSP with exponential on-off traffic (static network, traffic rate=20pkt/sec) . . . 102
74 Network lifetime of GSP and T-GSP with different exponential on-off traffic (pause time=0) . . . 103
75 Network lifetime of GSP and T-GSP with different exponential on-off traffic (static network) . . . 103
76 Layer structure to combine A-GSP, B-GSP and T-GSP . . . 106

1.0 INTRODUCTION

Recent advancements in wireless technologies have created a proliferation of wireless devices, including cellular phones, personal digital assistants (PDAs), laptops, pagers, and microsensors, enabling the development of ad hoc networks. An important challenge in the design of these networks is the limited energy of the wireless nodes in a possible brutal working environment. These restrictions require innovative communication techniques and protocols to increase the lifetime of the network. We investigate this problem and propose our research work. This chapter provides an introduction and a background for our work. In the following sections of this chapter, we provide an overview of ad hoc networks and ad hoc routing in Section 1.1 and 1.2. An overview of sensor network routing is given in Section 1.3. In Section 1.4, we introduce the related issues and general design guidelines of energy conservation in wireless ad hoc networks. The problem statement is given in Section 1.5.

1.1 AD HOC NETWORK OVERVIEW

There are currently two variations of mobile wireless networks. The first is known as infrastructured networks (i.e., networks with fixed wireless gateways connecting to a wired network). The bridges for these networks are known as base stations. Typical applications of this type of network architecture are cellular phone networks and infrastructure mode wireless local area networks (WLANs). An infrastructured network is suitable for locations where base stations can be placed. The advantage is that existing wireline networks can be leveraged to support access for mobile users without modifications to the network's control structure. The disadvantage is that it requires a fixed infrastructure - constraining node mobility, limiting network deployability, and in-

creasing installation and management costs. The second type of mobile wireless networks are the infrastructureless mobile networks, commonly known as ad hoc networks. In an ad hoc network, mobile nodes communicate with each other using multihop wireless links. There is no stationary infrastructure, and each node acts as a router, forwarding data packets for other nodes. Such networks have been studied in the past in relation to defense research, often under the name of packet radio networks [33]. Recently there has been a renewed interest in this field due to the availability of low-cost laptops and palmtops with radio interfaces. A mobile ad hoc networking (MANET) working group [35] has been formed within the Internet Engineering Task Force (IETF) to develop a routing framework for ad hoc networks. Some examples of possible applications of ad hoc networks include mobile computer users gathering for a conference, family members taking wireless computers home from their offices and schools without worrying about topologically related IP addresses, emergency disaster relief personnel coordinating efforts after a hurricane or earthquake, personal area network (PAN) with wireless devices that are closely associated with a single person and interactions between several PANs when people meet, wireless sensor networks in certain dangerous area, and soldiers relaying information for situational awareness on the battlefield.

1.2 AD HOC NETWORK ROUTING

Ad hoc networks inherit the traditional problems of wireless and mobile communications, such as bandwidth constraints and variable link capacity, power control and transmission quality enhancement. In addition, the multihop nature and the lack of fixed infrastructure introduce new research problems such as configuration advertising, discovery and maintenance, as well as ad hoc addressing and self-routing. A central challenge in the design of ad hoc networks is the development of dynamic routing protocols that can efficiently find routes between two communicating nodes. The routing protocol must be able to keep up with the high degree of node mobility that often changes the network topology drastically and unpredictably.

Traditional routing protocols, such as distance vector and link state protocols, were designed for static infrastructure based networks, and a dynamic topology was not considered in their design. For example, those routing protocols periodically emit control messages to exchange routing

information. However, in a large network with frequent changes of connections, connection quality and participants, this information becomes invalid soon. In other words, the mobility results in large number of control messages and to keep up with the mobile topology, the query period should be frequent. These are all negative factors for mobile nodes because they require radio transmissions and receptions, even during the idle period, which all drain energy from the battery. In addition, they consume the limited wireless bandwidth. Also, conventional routing protocols assume that routes are bi-directional and equal in quality, which is not always the case in ad-hoc networks.

Many different protocols have been proposed to solve the multihop routing problem in ad hoc networks. Basically, there are two approaches in providing ad hoc network connectivity: *topology-based* and *position-based* routing. Position-based routing algorithms require that information about the physical position of the participating nodes be available. Commonly, this position information can be achieved through the use of GPS or some other positioning services. The routing decision is based on the destination's position contained in the packet and the position of the forwarding node's neighbors, thus establishment and maintenance of routes are not required. Position-based routing requires mobile nodes to maintain the location information of themselves, neighbors and destinations. It's not trivial to maintain every other node's location or achieve a destination's location when needed. Additionally, position-based routing doesn't work in the situation where GPS is not available, e.g., inside a building. A survey and comparison of position-based approaches can be found in [34]. The proposed work in this dissertation is based on topology-based routing protocols, which use the information about the links that exist in the network to perform packet forwarding and we present a review in Chapter2.

1.3 SENSOR NETWORK ROUTING

Wireless sensor networks can be looked as ad hoc networks designed for special applications with special requirements. Normally, they require no or low mobility support. A sensor network consists of a large number of deployed sensor nodes [2]. The position of the sensor nodes is usually not predetermined, as the network may be deployed in inaccessible terrains or disaster relief oper-

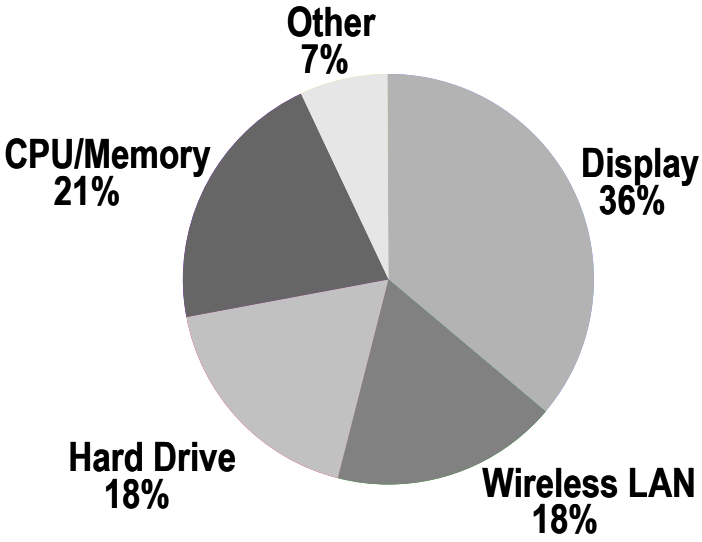

Figure 1: Power consumption by each subsystem of a Toshiba 410 CDT mobile computer (Pentium 90 with 8MBytes of EDO RAM and AT&T WaveLAN PC Card)

ations, resulting in a random topology.

Although many routing protocols have been proposed for wireless ad hoc networks, they are not necessarily appropriate for sensor networks. Sensor networks normally have larger size, higher density, more limited power supply and computational capacity than nodes in mobile ad hoc networks. Additionally, sensor networks can be characterized as data centric network, where users are interested in querying an attribute of the phenomenon, rather than querying an individual node. Furthermore, sensor networks are application-specific in that the requirements on the network change with the applications. As an example, in some applications the sensor nodes are fixed, but other applications require a combination of fixed and mobile nodes thus requiring mobility support. Also, adjacent nodes might have similar data; therefore, sensor networks should be able to aggregate similar data to reduce unnecessary transmissions and save energy. Lastly, assigning unique IDs may not be suitable in sensor networks because these networks are data centric – routing to and from a specific node may not be required. In addition, the large number of nodes will require long IDs, creating large overhead compared to data being transmitted.

1.4 ENERGY CONSERVATION

Since most wireless nodes in ad hoc networks are not connected to a power supply and battery replacement may be difficult, optimizing energy consumption in these networks has high priority, in order to prolong the network lifetime, during which the network can function properly. This requires not only energy efficient hardware, but also energy efficient protocols. Therefore, power management is one of the most challenging problems in ad hoc networking. Studies have shown that the significant consumers of power in a typical laptop are the CPU, memory, display, hard disk, keyboard/mouse, and wireless network interface card. Fig. 1 shows a typical example reported in [60]. As we can see, wireless communication accounts for a significant part of the energy consumption. Therefore, energy conservation needs to be considered not only in the hardware design, but also the network protocols.

In general, radios in an ad hoc network node can operate in four distinct modes of operation: *transmit*, *receive*, *idle*, and *sleep* [47] [14] [15]. Transmit and receive modes are for transmitting and receiving data. In the idle mode, the radio can switch to transmit or receive mode. Idle is the default mode for an ad hoc environment. The sleep mode has extremely low power consumption. An important observation reported in the literature is that the idle mode consumes almost same energy as receive mode [61] [47] [14]. Therefore, taking advantage of the sleep mode is very important in energy efficient protocols.

At the application layer, the energy conservation strategies take advantage of usage patterns associated with activities such as email retrieval and web browsing. The on-demand concept proposed in [64] can be implemented at the application layer, in which routing nodes predict the incoming packets based on the traffic pattern and determine when and how long to turn off the radio. Another application level strategy is the data aggregation [25] employed in the ad hoc sensor network. In a sensor network, adjacent nodes might have similar data. Data coming from multiple sensor nodes can be aggregated if they report similar data on the phenomenon observed when they reach a common routing node on the way back to the user/sink. This requires the intermediate nodes hand the packets up to the application layer.

A detailed analytical study of the energy efficiency of MAC layer protocols, including IEEE 802.11, is presented in [9]. Some of the general energy conservation guidelines for the MAC

protocol design include [32]:

- Collisions should be eliminated as much as possible since they result in retransmissions, which lead to unnecessary energy consumption and possibly unbounded delays.
- Since it may not be possible to fully eliminate collisions in an ad hoc network due to the mobility, a small packet size for registration and bandwidth request is preferred to reduce energy consumption.
- The receiver remaining on at all times results in significant power consumption. Turning off the transceiver by a schedule or whenever the node wants can conserve energy.
- Switching from transmit to receive modes and vice versa consumes significant time and power. If possible, the mobile should be allocated contiguous slots for transmission and reception to reduce such turnaround, thereby reducing the energy consumption.

The motivation of this dissertation is the need to provide energy conservation strategies for the ad hoc routing layer. Traditional metrics used to evaluate routing protocols include shortest-hop, minimum delay, and link stability. However, simply inheriting these metrics in ad hoc routing protocols may impose negative effects on the network performance because:

- To achieve the traditional metrics, the conventional ad hoc routing protocols require all the nodes in the network awake and keep listening. In ad hoc networks, due to the possible high density of the mobile nodes, routing can be done by a subset of the nodes and the rest can stay in the sleep mode to save energy.
- The resulting path may not be the most energy efficient one in terms of the total energy consumption for transmitting and receiving the packet by all the nodes along the path.
- The energy resource of a small set of nodes may be overused, which results in a reduced lifetime for those nodes, and hence the network lifetime due to network partition. Therefore, energy efficient routing can be achieved by either establishing routes in a way so that all the nodes deplete their battery power equally or avoiding routing through nodes with lower battery power.
- The periodic routing updates and the flooding overhead in the conventional ad hoc routing protocols consume lots of energy and should be reduced.

1.5 THE PROBLEM STATEMENT

In this thesis, we address the problem of the energy conservation in ad hoc networks. We focus on the applications that require a very low network cost with extremely constrained resources, e.g., a wireless microsensor network. Therefore, an energy conservation technique with very low overhead is preferred. We introduce a sleep mode into the network and propose a new sleep management protocol, which schedules the working/sleep cycle of every node. The objective is to reduce energy consumption while maintaining the connectivity of the network, so that the network lifetime can be prolonged without sacrificing the network performance in terms of network throughput and end-to-end packet delay. We expect that the proposed protocol can support both sensor networks, where there is no mobility, and ad hoc networks in certain scenarios. An analysis of energy efficiency will be conducted for the case without mobility. More detailed studies of the network performance with and without mobility will be done via simulation. We will investigate the effects of parameter changes and the impacts of routing protocols on our protocol. Finally, we will extend our work to different network scenarios, and improve the protocol performance by making the protocol adapt to the traffic and network parameters.

1.6 ORGANIZATION

This dissertation is organized as follows. Chapter 2 reviews wireless ad hoc network routing protocols and existing energy efficiency techniques. Drawbacks and unaddressed issues of existing work are identified to fit in our proposed research. Chapter 3 presents our sleep management approach based on the percolation theory, or gossiping. Two versions of the scheme are proposed, i.e., *synchronized* and *asynchronous*, which use probabilistic information to manage the radio modes of the nodes. The objective is to prolong the network lifetime while maintaining the network connectivity without major performance degradation. We also conduct an analytical study of the approach to show its correctness and effectiveness in this Chapter. Chapter 4 presents a comprehensive simulation study on the proposed energy conservation scheme to justify the analytical results and investigate the effects of parameter tuning. The impacts of different routing protocols

on the approach are also discussed. The improvement and extensions of our work are presented in Chapter 5, so that our work can apply to a more realistic network. The contributions and future research directions are given in Chapter 6.

2.0 LITERATURE REVIEW

This chapter reviews a set of the protocols related to the proposed research. We first introduce different types of topology-based ad hoc routing protocols in Section 2.1 and conventional sensor network routing protocols in Section 2.2. Secondly, in Section 2.3 we will concentrate on the energy conservation strategies for routing in the literature. They are classified into metric-based and sleep-based protocols. We discuss the performance trade-offs and unaddressed issues of the existing research work.

2.1 AD HOC NETWORK ROUTING

Topology-based ad hoc routing can be divided into *flat* routing and *cluster-based (hierarchical)* routing based on the organizational structure of the network. In the flat-routed architecture, all the nodes are equal and packet routing is done based on peer-to-peer connections. However, in hierarchical networks, at least one node in each lower layer is designed to serve as a gateway or coordinator to higher layers. Flat routing can be further divided into *proactive*, *reactive* and *hybrid* approaches.

2.1.1 Proactive Routing

Proactive routing protocols attempt to maintain consistent, up-to-date routing information from each node to every other node in the network. Such protocols are termed proactive because they store route information even before it is needed. They are also called table-driven [50] because these protocols require each node to maintain one or more tables to store routing information.

Keeping track of routes for all destinations in the ad hoc network has the advantage that the communication with an arbitrary destination experiences minimal initial delay from the point of view of the application. When the application starts, a route can be immediately available in the routing table. However, proactive protocols suffer the disadvantage of additional control traffic that is needed to continually update stale routing entries. An ad hoc network is presumed to contain numerous mobile nodes. Therefore, routes are likely to be broken frequently. On one hand, much control information is needed to repair the broken route, on the other hand, most of the routing entries expires before applications use them. Proactive routing protocols can be classified into *distance vector routing* and *link state routing* based on the type of routing information.

In distance vector (DV) routing, each node monitors the cost of its outgoing links and periodically broadcasts, to each of its neighbors, its current shortest distance to every other node in the network, estimated based on the information gathered from its neighbors. Compared to the link state (LS) method, DV is computationally more efficient, easier to implement and requires much less storage space. However, this algorithm can cause the formation of both short-lived and long-lived loops. Destination Sequenced Distance Vector Routing Protocol (DSDV) [43] and Wireless Routing Protocol (WRP) [40] are two distance vector routing protocols proposed for ad hoc networks.

In link state (LS) routing, each node maintains a view of the network topology with a cost for each link. To keep these views consistent, each node periodically broadcasts the link costs of its outgoing links to all other nodes using a protocol such as flooding. As a node receives this information, it updates its view of the network topology and applies a shortest path algorithm to choose its next hop for each destination. LS is attractive because it can avoid long-term loops, construct multiple paths, achieve faster convergence, and more easily support QoS routing decisions. However, the flooding overhead of LS and the impact on the MAC layer due to the frequent transmission of small link state packets is not acceptable for an ad hoc network. To address these issues, a number of approaches are proposed in the literature. These approaches can be classified into *efficient dissemination* and *limited dissemination* approaches. In efficient dissemination, updates are sent throughout the network, but more efficiently compared to traditional flooding. Two examples are OLSR [11] and TBRPF [3]. In OLSR, each node selects a set of Multiplepoint Relay (MPR) points in its one hop neighborhood to retransmit its broadcast messages. This set is selected such

that it covers all nodes that are two hops away. TBRPF uses the concept of reverse-path forwarding (RPF) to broadcast link state updates in the reverse direction along the spanning tree formed by the minimum-hop paths from all nodes to the source of the update. Limited dissemination attempts to reduce the routing update overhead by restricting the scope of routing updates in space and time. Typical examples include FSR [28], STAR [16], FSLS [51], etc. FSR exchanges the link information between adjacent nodes only and uses "Fisheye" technique to reduce the size of the update message by updating the network information for nearby nodes at a higher frequency for better accuracy. Similarly, in STAR, a node sends updates to its neighbors only. The update is regarding the links along the preferred paths from a source to each desired destination, which constitute a source tree. FSLS reduces the frequency of link state updates (LSU) propagated to distant nodes based on the observation that in hop-by-hop routing, changes experienced by nodes far away tend to have little impact on a node's 'local' next hop decision. A node wakes up every $2^{i-1} \times t_e (i = 1, 2, 3, ...)$ seconds and transmits a LSU with Time To Live (TTL) set to s_i if there has been a link status change in the last 2^{i-1} seconds. Different approaches may be implemented by considering different $\{s_i\}$ sequences.

2.1.2 Reactive Routing

Reactive routing creates and maintains routes only when desired by the source node. Therefore, it's also known as on-demand, source-initiated, or demand-driven routing [50]. When a node requires a route to a destination, it initiates a route discovery process within the network, typically, by some form of flooding. This process is completed once a route is found or all possible route permutations have been examined. Once a route has been established, it is maintained by a route maintenance procedure until either the destination becomes inaccessible along every path from the source or until the route is no longer desired. Compared to proactive routing, reactive routing consumes far less bandwidth for maintaining the routing tables at each node when only a small subset of all available routes is in use at any time. However, reactive has some inherent limitations. First, since the routes are maintained on-demand, most applications are likely to suffer a long delay when they start. Second, when the topology of the network changes frequently, route maintenance may generate a significant amount of network traffic. Third, packets en route to the destination are

likely to be lost if the route to the destination changes. Finally, it's not easy for reactive routing protocols to support QoS. Different reactive strategies have been proposed with the following two being the most popular ones, namely: *source routing* and *hop-by-hop routing*.

In source routing, to send a packet to another node, the sender constructs a source route in the packet's header, giving the address of each node in the network through which the packet should be forwarded to reach the destination. The sender then transmits the packet to the first hop identified in the source route. When a node receives a packet, if this node is not the final destination of the packet, it simply transmits the packet to the next hop identified in the source route in the packet's header. The Dynamic Source Routing (DSR)[29][30][31] protocol employs this concept and maintains a route cache in each node to contain the source routes the node is aware at. As an improvement, Neighborhood aware Source Routing (NSR) [58] reduces the effort required to fix source routes locally by using alternate links available in the two-hop neighborhood of nodes. The two-hop neighborhood information is maintained by exchanging link-state information among neighboring nodes.

Source routing has many advantages, including simplicity, correctness, and flexibility [26]. Since all routing decisions for a packet are made by the sender, intermediate nodes do not need to maintain up-to-date, consistent routing information for the destination. By including the source route in the packet header, the route over which a packet is forwarded can be guaranteed to be loop free. In addition, for reasons such as load balancing, the perceived longevity and reliability of the route, the security of the nodes, or differentiated treatment of different types or classes of packets for QoS, it is possible for the sender to use different routes for different packets, without requiring coordination or explicit support by the intermediate nodes. The disadvantage of source routing is that each packet contains considerable overhead because the entire route must be recorded in the packet header, thus decreasing the bandwidth available for data, increasing the transmission latency of each packet, consuming extra battery power and source routing is not scalable. To reduce this extra per-packet overhead, [26] proposed *implicit* source routing, which is similar to the techniques used for MPLS or ATM virtual paths. Each packet is tagged with a *flow identifier* when the packet is sent. Intermediate nodes retain the information indicating the next hop to which packets belonging to that flow should be forwarded. Although this makes it look like hop-by-hop routing as discussed below, implicit source routing still keeps the basic operations and therefore

the important properties of source routing.

Unlike source routing, where overhead is incurred by carrying the entire source routes in each data packet, hop-by-hop routing relies on dynamically establishing route table entries at intermediate nodes. The advantage of hop-by-hop routing is that every packet carries only the destination address instead of full routing information as in source routing. The memory overhead is slightly higher in source routing because of the need to remember full routes, as opposed to only next hop information in hop-by-hop routing. Therefore, the hop-by-hop routing is more scalable. The disadvantage of hop-by-hop routing is the overhead and operation required to maintain routing tables and prevent loops. In addition, in case of link failure, the source has to reinitiate a new path discovery process, since there is no route cache to take advantage of. The Ad-hoc On-demand Distance Vector (AODV) [44] [45] and Source-tree On-demand Adaptive Routing (SOAR) [49] are two hop-by-hop routing protocols.

2.1.3 Hybrid Routing

As discussed earlier, the proactive ad hoc routing approach relies on an underlying routing table update mechanism that involves the constant propagation of routing information. In contrast in the reactive approach, a node has to wait until a route can be discovered when it desires to communicate with a destination. The major difference between these two approaches is that proactive protocols provide routes with less delay at the cost of more routing overhead, but reactive protocols provide routes with less routing overhead at the cost of more delay. Some work has been done attempting to combine the advantages of two approaches, called hybrid routing. The hybrid approaches reviewed in this section are based on flat organization structure and integrate proactive and reactive approaches for the whole network. Hierarchical hybrid approaches will be presented in the next section.

One reasonable middle point between proactive and reactive protocols might be to keep track of multiple routes between a source and a destination node. The idea of discovering alternative path before an active path breaks results in adding proactive route selection and maintenance to reactive ad hoc routing algorithms. More specifically, when a path is likely to be broken, a warning is sent to the source indicating the likelihood of a disconnection. The source can then initiate

path discovery early, potentially avoiding the disconnection altogether. This is *preemptive routing* proposed in [17]. With pure reactive routing, when a path break occurs, the connectivity of the flow is interrupted and a hand-off delay is experienced by the packets that are ready to be sent. This increases both the average and variance (jitter) of packet latency. Preemptive routing switches to an alternative good path *before* a break, minimizing both the latency and jitter. The signal strength is used to trigger path discovery. Mitigation of channel fading and other transient interferences can be achieved by the mechanisms used to solve this problem in the cellular systems. Other warning criteria such as location/velocity and congestion can also be used as the preemptive trigger. This scheme may increase the routing overhead of reactive protocols since some path discoveries are being carried out proactively. Also, better mobility models and the techniques for selecting the trigger criteria are needed to improve the performance.

Instead of adding proactive route maintenance to on-demand algorithms, Adaptive Distance Vector (ADV) routing algorithm [4] reduces the periodic routing overhead of proactive protocols by using some on-demand characteristics. ADV starts with a basic distance vector algorithm and varies the frequency and the size of the routing updates in response to the network load and mobility conditions. First, it maintains the routes only to active receivers to reduce the number of entries advertised. A node is an active node if it is the receiver of any currently active connection. Secondly, ADV adaptively triggers partial and full routing updates such that periodic full updates used in basic DV algorithms are obviated. ADV monitors the node mobility by keeping track of the number of neighbor changes in a time period, during which a fixed number of full updates are triggered. Traffic load is measured with the number of packets buffered for lack of route. The information is stored in routing table for each entry to encourage active nodes to advertise routes more frequently and, at the same time, discourage the non-receiver nodes from transmitting more than necessary updates. Unlike proactive DV-based protocols, a node in ADV does not trigger an update whenever it sees a change in the metric for a routing entry, nor like in reactive protocols, in ADV the need for a fresh valid route to an active receiver does not immediately result in a route discovery process. In ADV, a fresh valid route can only be obtained from neighbor updates, so obtaining a valid route could take a long time in ADV.

2.1.4 Cluster-based Routing

It is well known that cluster-based or hierarchical routing is essential for achieving scalability in fixed infrastructure networks. However, the clusters of an ad hoc network must be maintained dynamically. This introduces a number of difficult challenges. This section presents a review of the literature that addressed the cluster-based routing methodology in wireless ad hoc networks.

Increasing a node's transmission range by increasing its transmission power enables direct communication with a more distant node to get reduced-hop backbone topologies, but it also increases interference since a node's transmissions will be received at higher power and by a larger number of nodes. Based on this observation, different cluster-based routing schemes and the associated clustering methodologies are proposed, i.e., non-adaptive clustering and adaptive clustering, hierarchical proactive routing and hybrid routing.

Non-adaptive clustering: Although cluster membership is maintained dynamically, cluster size is not adaptive either to node mobility, transmission characteristics or network traffic. However, the advantage is its simplicity.

Several protocols employed non-adaptive cluster formation. In Zone Routing Protocol (ZRP) [20] [21] [42], each node maintains its own routing zone, a cluster of nodes that can be reached along the paths that are no longer than ρ hops, where ρ is defined as the zone radius. Since each node is the center of its own routing zone, the zone topology consists of a set of overlapping dynamic clusters. Multimedia support for Mobile Wireless Networks (MMWN) [48] organizes clusters according to a set of parameters that control the size of each cluster and the number of hierarchical levels. The predefined static parameters cannot handle node mobility very well.

Adaptive clustering: In [38] [37] the (α, t) cluster-framework is designed to sense and adapt dynamically to changing environments. Clusters adapt dynamically based on the mobility characteristics of the local nodes so that the probability of path failure due to node movement can be bounded over time. The basic idea of the adaptive clustering is to partition the network into clusters of nodes that are mutually reachable along cluster internal paths that are expected to be available for a period of time t with a probability of at least α. The cluster formation in Dynamic Group Routing (DGR) [7] is adaptive to the changing topology by taking in account the node connectivity information or node degree. A node whose degree is greater than most of its neighbors is called

a *positive* node, otherwise a *non-positive* node. Connected positive nodes form a positive cluster and connected non-positive nodes form a non-positive cluster. A *routing group* is formed by one positive cluster with its adjacent non-positive clusters. Adaptive clustering is able to maintain an effective topology that adapts to node mobility and changing topology. Therefore, routing can be more responsive and optimal when mobility rates are low and more efficient when they are high as in (α, t) cluster-framework, or route maintenance more efficient and less overhead as in DGR.

Hierarchical proactive routing: For large clusters, intracluster routing is needed and proactive routing is easier to maintain within the clusters. Clustered Spine Routing (CSR) [12] employs spine routing [57] within the clusters and link state routing at the cluster level. Proactively maintaining routing for both intra and intercluster routing is not scalable, even storing network state in spine nodes only, as it's not trivial to maintain spine and cluster membership tables at all the roots in a highly dynamic environment. In MMWN, nodes proactively maintain routes to all destinations in the same cluster using a link state protocol and source routing to setup connections. Border nodes are dynamically arranged into virtual gateways (VG), which provide the routes to remote destinations. Higher level clusters use VGs and virtual links (VL) between VGs in place of physical nodes and links to proactively maintain the cluster topology. Due to service aggregation at each level of the clustering hierarchy, a higher level VL is expected to be more stable than a lower level VL in most applications. But in order to support proactive maintenance, the system suffers substantial overhead from the need to maintain the mobility management, which is handled by a route request process in reactive algorithms.

Hybrid routing: Hybrid cluster-based routing attempts to balance the tradeoff between proactive and reactive routing algorithms, typically by using proactive routing within the clusters and reactive routing for remote clusters.

In Zone Routing Protocol (ZRP) [20] [21] [42], Intra-zone (cluster) routing (IARP) [21] consists of a proactive link state protocol that maintains a hop-count limited routing table. Inter-zone routing (IERP) [21] is managed by a reactive routing process. Variants of some proactive link state protocols and reactive protocols can be employed. The inter-zone routes that are established are flat, hop-by-hop routes, which must be dynamically restored whenever node mobility leads to link failures along active paths. The fully overlapping feature of ZRP introduces serious complexities in managing the route search process as query explosion may happen. Similar to ZRP, in (α, t)

cluster-framework [38] [37], intracluster routing uses a proactive strategy, which can be implemented with any distributed proactive routing. The Inter-Cluster Routing Protocol (ICRP) [37] proposed does not require paths to be routed through clusterheads or gateway nodes in each cluster. If the destination is not in its own cluster, the source multicasts a query message to relay nodes, which are in the adjacent clusters and connected with the nodes in the source's own cluster. The relay nodes, recursively, forward this message to their own relay nodes, until the destination is reached. The route is a sequence of these relay nodes and valid as long as each relay node remains in its cluster and clusters are connected. In Dynamic Group Routing (DGR) [7], positive nodes maintain the topology of its positive cluster and adjacent non-positive clusters, and non-positive nodes maintain only the topology of its non-positive cluster and the links between its cluster and adjacent positive clusters. Therefore, routing is maintained proactively within a cluster, proactively for positive nodes and reactively for non-positive nodes within a group, reactively for remote nodes.

From the ad hoc routing protocols reviewed in this section, we can see that, in summary, proactive routing attempts to maintain consistent, up-to-date routing information from each node to every other node in the network. It suffers the disadvantage of additional control traffic that is needed to continually update stale routing entries. Reactive routing creates and maintains routes only when desired by the source node. Since the routes are maintained on-demand, most applications are likely to suffer a long delay when they start. Hybrid routing tries to tradeoff the advantages and the disadvantages of these two approaches, while the cluster-based routing is proposed to achieve scalability.

2.2 SENSOR NETWORK ROUTING

In this section, we review the conventional wireless sensor network routing protocols which do not consider energy conservation directly. They may achieve energy efficiency in an indirect way, e.g., by reduced routing overhead. We can classify them into *flooding*, *forwarding*, and *data-centric* based routing depending on their packet forwarding techniques.

2.2.1 Flooding

Flooding is an old technique that can be used in sensor network. In flooding, every node repeats the data once by broadcasting. It doesn't require costly topology maintenance and complex route discovery algorithms. But it has several deficiencies as follows [2]:

- Implosion: duplicated messages are sent to the same node. A node with multiple neighbors may get multiple copies of the same message.
- Overlap: if two sensors share the same observation region, both of them may sense the same stimuli at the same time. As a result, neighbor nodes receive duplicated messages.
- Resource blindness: flooding doesn't take into account the available resources.

2.2.2 Forwarding

To overcome the problems of flooding, forwarding schemes utilize certain local information to forward messages. Unlike traditional routing protocols, forwarding doesn't maintain end-to-end routing information. Instead, intermediate nodes just maintain the neighbor information. Different information is used by different routing protocols, as we discuss in the rest of this section.

In Gossiping [23], a gossiping node only forwards data to one randomly chosen neighbor, so it doesn't maintain any information or we can say it uses randomness to forward data. Since there is only one copy of the data in the network at a given time, there is no implosion in gossiping. But overlap can not be avoided. Another drawback of this scheme is that data propagation is slow and can not be bounded.

Best Effort Geographical Routing Protocol (BEGHR) [41] employs position information to forward data, so it needs GPS or another positioning service. Each node oscillates between client mode and server mode. In client mode, the node forwards data to the follow on node. In server mode, the node receives data. A node allocates different time periods for these two modes. Basically, the node closer to the home node will stay in server mode longer since it may have more data to receive. Since every node knows its own position and the home node's position, data can easily avoid a loop. The probability of data propagation time can be bounded. But data propagation is not guaranteed, there is no way to find a packet loss. In addition, this scheme doesn't support mobility, since mobility may cause the node to change the time period to remain in client and server mode

frequently, therefore affect the performance of the network. Finally, the GPS or other positioning system is an extra cost to the network.

Field based Optimal Forwarding [62] employs a *cost field* to forward data. A cost field is the minimum cost from a node to the sink on the optimal path. A backoff-based cost field establishment algorithm has been proposed to create the cost field in each node. The basic idea is each node backs off some time after receiving the cost field value of one of its neighbors to wait for the optimal value so that it just needs to broadcast its own cost field once. After the sink broadcasts an ADV (advertisement) message containing its own cost (0 initially), the message propagates throughout the network. An intermediate node sets its cost as the sum of the cost of the link from which it receives the message and the cost in the message, and the deferral time to be proportional to the link cost. The timer is reset if another copy of the ADV message from other links generates a smaller deferral time. This backoff-based algorithm always sets up the optimal cost field with one message (containing optimal cost) broadcast at each node [62]. Every data packet takes the cost field value of the source and the cost consumed so far. A node forwards this packet only if the sum of the consumed cost and the cost at this node matches the source's cost. Therefore, the packet is forwarded along the optimal path. This scheme doesn't require the intermediate nodes to maintain explicit "forwarding path" status, even node ID. The time and space complexities are constant, so it scales to the large network. The drawback of this scheme is the frequent cost field refreshing in case of mobility.

2.2.3 Data-Centric

In data-centric based routing, an interest message is disseminated to assign the sensing tasks to the sensor nodes, so attribute-based naming [2] is required. The users are interested in querying an attribute of the phenomenon, rather than querying an individual node. So instead of using "the temperature read by a certain node", we may use "the areas where the temperature is over $70^{\circ}F$". Also, data aggregation is used to solve the implosion and overlap problems.

There are two types of data-centric based routing, based on either the sink node broadcasting the interest message for data or the sensor nodes broadcasting an advertisement for the available data and waiting for a request.

In directed diffusion [27], the sink sends out the interest as a task description, to the network. After receiving an interest from a neighbor, a node sets up a gradient to that neighbor associated with a data rate required by the interest message. After the source node receives the interest, data is sent along the gradient paths. At the beginning, there may be multiple paths and multiple copies are sent to the sink. The sink can reinforce some paths by repeating the interest message for more recent data. If there are multiple sinks or sources, data aggregation can be achieved. Intermediate nodes can pick the highest data rate for the same interest.

In directed diffusion, there is no need to maintain globally unique node identification. Each node just needs to maintain neighbor nodes information. A path loop can be avoided by comparing with previous received data. Multiple path transmission makes this scheme robust. But it's not energy efficient, because of the multiple paths and the sub-optimal paths cased by local information maintenance.

Sensor Protocols for Information via Negotiation (SPIN) [25] is used to disseminate individual sensor's observation to all the sensors in the network. It's based on the observation that "sensor nodes operate more efficiently and conserve energy by sending data that describe the sensor data instead of sending all the data". It has two components negotiation and resource management, to overcome the deficiencies of flooding, implosion, overlap, and resource blindness. With negotiation, nodes negotiate with each other before transmitting data, so only useful information will be transferred. With resource management, nodes monitor their resource consumption to cut back on certain activities when energy is low, like forwarding other nodes' data. During the negotiation, if node A has new data to send. An advertise message (ADV) is sent to its neighbor, say node B. If node B is interested in the data, it replies a request message (REQ) to ask for the data. Then node A sends the data to node B and node B repeats this process. If some of its neighbors are not interested in the data, node B will not receive the REQ messages from them and it is not necessary to send to them. In this process, data aggregation can be implemented. For example, if node B has its own data, it can aggregate with node A's data and send an ADV message for this aggregated data.

SPIN is simple and no end-to-end routing information is needed. Data aggregation is used to reduce implosion and overlap problem. But it's designed for lossless networks. Although it can be modified to adapt to lossy networks by re-sending control messages, the cost will be high. The

basic idea of spin is to use smaller control message to describe data to avoid implosion and overlap, so if we need multiple control messages, the benefit may be reduced. Also, it disseminates data to all sensors in the network, not to a single sink.

2.3 ENERGY EFFICIENT ROUTING

The battery lifetime of mobile nodes significantly impacts the performance of ad hoc networks. Since nodes in ad hoc networks need to relay their packets through other nodes, a decrease in the number of nodes may partition the network. The sources of energy consumption, with regard to network operations, can be classified into two types [32]: communication related and computation related. Computation is chiefly concerned with protocol processing aspects. It mainly involves usage of the CPU and main memory and, to a very small extent, the disk or other components. Communication involves usage of the transceiver at the source, destination and intermediate nodes. The goal of energy efficient routing protocols is to optimize the transceiver usage for a given communication task with limited energy resources. There exists a tradeoff between computation and communication costs. Schemes that strive to achieve lower communication costs may result in higher computation needs, and vice versa. For instance, data compression and data aggregation [25] used in some sensor network protocols reduce energy cost of communication but increase that of computation. Hence, energy conservation techniques should attempt to strike a balance between the two costs. This section presents a review of energy efficient routing techniques. Basically, they can be classified into two categories: *metric-based* and *sleep-based*.

2.3.1 Metric-based Routing

To prolong the lifetime of each node, metric-based energy efficient routing protocols consider power consumption and select the best path to minimize the total power needed and maximize the lifetime of all nodes. There are four variations of route selection schemes to achieve one or both of these goals [59], namely Minimum Total Transmission Power Routing (MTPR), Minimum Battery Cost Routing (MBCR), Min–Max Battery Cost Routing (MMBCR) and Conditional Max–

Min Battery Capacity Routing (CMMBCR).

MTPR selects the path with the minimum total transmission power along it, which is the sum of the transmission power of all links on the path. A modified Dijkstra's shortest path algorithm presented in [54] employs this scheme. Under light traffic load, MTPR selects the same route as shortest-hop routing if the energy consumption for every hop is roughly same. On the other hand, if some nodes are heavily loaded, MTPR may use a non-shortest-hop path to avoid energy consumption on contention. MTPR has a critical disadvantage, although it can reduce the total power consumption of the overall network, it does not reflect directly on the lifetime of each node. If the best routes are via a specific node, the battery of this node will be exhausted quickly and die soon.

MBCR uses the remaining battery capacity of each host as a metric to describe the lifetime of each node and selects the route with the maximum total remaining battery capacity. This metric prevents hosts from being overused, thereby increasing their lifetime and the time until the network is partitioned [56]. However, since only summation of remaining battery capacity is considered, a node with little remaining battery capacity in a route with maximum total remaining battery capacity can still be overused.

To make sure that no node will be overused, MMBCR always avoids the route with nodes having the least battery capacity among all nodes in all possible routes. The battery of each node is used more fairly than in previous schemes. But it does not guarantee that the minimum total transmission power paths will be selected under all circumstances, therefore it reduces the lifetime of all nodes. The heuristic algorithms presented in [5] employ this idea. The paper models the problem as a linear programming problem, in which the objective is to maximize the lifetime of the network, i.e., the time until the first battery drains out. The distributed Bellman-Ford algorithm is modified to find the minimum cost path. Multiple power levels are considered in the model and the work has been extended to the case of multicommodity flow in [6].

In CMMBCR, when all nodes in some possible routes between a source and a destination have sufficient remaining battery capacity (i.e., above a threshold), a route with minimum total transmission power is chosen. However, if all routes have nodes with low battery capacity (i.e., below a threshold), routes including nodes with the lowest battery capacity are avoided [59]. CMMBCR can maximize the lifetime of each node and use the battery fairly, which cannot be achieved simul-

taneously by applying the other three schemes. A similar idea is used in [52] for wireless sensor networks. Each node has a *height* to indicate its cost to the user. A *gradient* is established for each link along the paths from the sources to the user (i.e. the sink) of the network, which is the difference between a node's height and that of its neighbor. When a node detects that its energy reserve has dropped below a certain threshold, it discourages others from sending data to it by increasing its height. The upstream nodes will update their own heights to keep all the gradients consistent, since the gradient is always positive.

The energy aware routing presented in [55] avoids overusing the lowest energy route by occasionally using sub-optimal paths to prevent the network from partition. To achieve this, multiple paths are found between a source and a destination, and each path is assigned a probability of being chosen, depending on the energy metric. Every time the data is to be sent from the source to the destination, one of the paths is randomly chosen depending on the probabilities. The energy metric used is $C_{ij} = e_{ij}^{\alpha} R_i^{\beta}$, where C_{ij} is the cost metric between node i and node j, e_{ij} is the energy used to transmit and receive on the link, R_i is the residual energy at node i normalized to the initial energy of the node. The weighting factors α and β can be chosen to find the minimum energy path or the path with nodes having the most residual energy or a combination of the above. The probability assigned to a path is inversely proportional to the cost.

2.3.2 Sleep-based Routing

Unlike metric-based schemes, sleep-based schemes are motivated by the observation that nodes in idle mode consume significant amount of energy, which is only slightly smaller than that in transmit and receive mode. For example, a measurement of the current and power supply for Lucent IEEE 802.11 WaveLan PC card from [14] is shown in Table 1 as well as the values in the specification. For clarity, we calculate the associated power consumptions for each mode and append it to Table 1. It clearly shows that putting a wireless node into receive or idle mode cannot significantly reduce energy consumption. Similar results have been reported in [61] [47] for other types of cards and wireless sensor devices. Therefore, sleep-based schemes turn off the radio and put them in sleep mode when it not in use in order to save energy more efficiently. In practice, on one hand, we want as many nodes as possible to turn their radio off to avoid idle mode energy

Table 1: Lucent IEEE 802.11 WaveLAN PC Card Characteristics

Radio mode ($2Mbps$)	Measured (mA)	Spec (mA)	Measured (W)	Spec (W)
Transmit	280	330	1.327	1.65
Receive	204	280	0.967	1.4
Idle	178	n/a	0.844	n/a
Sleep	14	9	0.066	0.045
Radio mode ($11Mbps$)	Measured (mA)	Spec (mA)	Measured (W)	Spec (W)
Transmit	284	280	1.35	1.4
Receive	190	180	0.90	0.9
Idle	156	n/a	0.74	n/a
Sleep	10	10	0.047	0.05
Power Supply	4.74 V	5 V	4.74 V	5 V

consumption. On the other hand, we also need enough nodes stay awake for the transmission between any source and destination without significant extra delay. Basically, there are two types of sleep-based energy efficient protocols in the literature: cluster-based and flat.

In cluster-based routing protocols, all nodes are organized into clusters with one node selected as the cluster-head for each cluster. Low-Energy Adaptive Clustering Hierarchy (LEACH) [24] is designed for proactive sensor networks, in which the nodes periodically switch on their sensors and transmitters, sense the environment and transmit the data. Nodes with more remaining energy are selected as the cluster-heads. The optimal number of clusters in the network is determined a priori depending on the network topology and the relative costs of computation versus communication. Non-cluster-head nodes choose a cluster to join based on the signal strength of the received advertisements from the cluster-heads and the symmetric propagation channels are assumed. The cluster-head assigns TDMA slots to its members to schedule the communication and the sleep mode. Nodes communicate with their cluster-heads directly and a randomized rotation of the cluster-heads is used to evenly distribute the energy load among the sensors. However, in LEACH, cluster-heads forward the data to the sink or the base station directly, which is not always energy efficient. Furthermore, overhead is generated to inform each other about their status, i.e., cluster-head and membership. Additionally, the optimal number of clusters in the network is determined a priori only by simulation instead of computation. At last, the network needs to be synchronized

to repeat the clustering periodically so that the randomized rotation of the cluster-heads can be implemented.

Threshold sensitive Energy Efficient sensor Network protocol (TEEN) [36] employs a similar hierarchical clustering scheme to LEACH and is designed for reactive networks, where the nodes react immediately to sudden changes in the environment. Nodes sense the environment continuously, but send the data to cluster-heads only when some predefined thresholds are reached. Adaptive Periodic Threshold sensitive Energy Efficient sensor Network protocol (APTEEN) protocol [1] combines the features of the above two protocols by modifying TEEN to employ a TDMA schedule similar to LEACH and make it send periodic data.

The cluster-heads in the cluster-based routing protocols can easily arrange the sleep mode of each member node to conserve energy. However, high complexity and overhead are incurred due to the clustering.

Span [8] forms a multi-hop forwarding backbone to provide about as much total capacity as the original network. The backbone nodes are termed *coordinators*. Other nodes can go to sleep more often to conserve their energy as they do not carry any traffic. HELLO messages are used periodically to exchange status and neighbor information. From these HELLO messages, each node constructs a list of the node's neighbors and coordinators, and for each neighbor, a list of its neighbors and coordinators. A non-coordinator node announces to be a coordinator if it discovers that two of its neighbors cannot reach each other either directly or via one or two coordinators. To reduce the announcement contention, a randomized backoff delay is employed, which takes two factors into account: the amount of remaining battery energy, and the number of pairs of neighbors the node can connect together. Therefore, less coordinators with more remaining energy are more likely to form the backbone. Backbone functionality is rotated among the nodes to balance the energy consumption. A node withdraws as a coordinator after a period of time, thus gives its neighbors a chance to become coordinators. Basically, the set of coordinators elected by Span is a connected dominating set of the network. A connected dominating set S of a graph G is a connected subgraph of G such that every u in G is either in S or adjacent to some v in S. Span tries to preserve the original capacity of the network by selecting the nodes who can provide more connectivity among their neighbors as the backbone nodes. Therefore, the coordinator set in Span is larger than a minimal connected dominating set. In case of low traffic load and high density

network, more nodes than necessary are active and consume energy. Additionally, extra routing overhead, i.e., HELLO message, is required.

Geographic Adaptive Fidelity (GAF) [61] conserves energy by identifying nodes equivalent from a routing perspective and turning off unnecessary nodes. The network is divided into small virtual grids based on the location information so that all nodes in the adjacent grids can communicate with each other directly. If a virtual grid is a square with r units on a side and R is the maximal transmission range of a node, then $r \leq \frac{R}{\sqrt{5}}$. In GAF, nodes are in one of three states: *sleeping*, *discovery*, *active*. When in state discovery, a node turns on its radio and exchanges discovery messages, which include grid and state information, to find other nodes within the same grid. A node in discovery and active states can switch to sleeping state when it is sure that some other node in the same grid will handle the routing. Therefore, at each point in time, only one node in each grid is active. GAF gives nodes with longer expected lifetime higher priority to be active and handle routing. An active node periodically broadcasts discovery messages. A node in the sleeping state wakes up after an application-dependent sleep time and switches back to the discovery state. To overcome mobility, the sleeping time is also dependent on the duration that the active node in its grid is expected to stay. In systems where movement is less predictable, this duration is difficult to estimate. GAF tries to maintain *routing fidelity*, which is defined as uninterrupted connectivity between communicating nodes. However, GPS or other positioning systems are required to get the location information for grid formation. Control overhead is generated to exchange the grid, neighbor and state information within a grid. Also a good mobility model is required to predict the departure of an active mobile node in a grid.

Sparse Topology and Energy Management (STEM) [53] exploits a separate paging channel to wake up nodes to trade off setup latency for energy savings. A low duty cycle radio is used to reduce the energy consumption of the paging channel. A node with outgoing packets sends out beacons to the target node on the paging channel. The paging radio on each node periodically wakes up to listen if any node is contacting it. In order for the target to receive at least one beacon, the wakeup time of a paging radio should be at least as long as the transmission time of a beacon plus the interarrival time of beacons. The dual channel system can reduce the interference between the paging and data transmission. In general, Span and GAF trade off network density for energy efficiency and STEM exploits time dimension rather than the density dimension. Therefore, the

path setup latency in STEM is long. In addition, STEM requires dual frequency radio and beacon packets to wake up a sleeping node, which increases the cost of a network.

An on-demand scheme is proposed in [64] to reduce energy consumption for on-demand ad hoc routing protocols. Connectivity is only maintained between pairs of senders and receivers and along the route of data communication and the nodes that do not carry any traffic can transit to power-save mode and be dispensed from consuming energy. In power-save mode, a node is sleeping most of the time and wakes up periodically to check for pending messages. The transition from power-save mode to active mode is triggered by communication events such as routing control messages or data packets. Different messages have different meanings for state transitions. For example, data packets and route reply messages in a reactive routing protocol are good hints to activate a node. Route request messages, however, are flooded throughout the network and provide little information. The transition from active mode to power-save mode is determined by a soft-sate timer. The timer is refreshed by the same communication events that trigger a transition to the active mode. A node keeps track of its neighbors' modes either by HELLO messages or by snooping transmissions over the air. With the on-demand concept, more energy can be conserved since only the nodes carrying traffic are awake. However, applications may face long initial delay because nodes must be woken up at the beginning of the transmission. Furthermore, the on-demand scheme requires the knowledge of the semantics of the control messages generated by the routing protocols, which makes the scheme difficult to implement.

2.3.3 Discussion

Although the existing research on ad hoc energy conservation has explored many methodologies, there has been no strategy perfect for all scenarios. Tradeoffs between many factors, e.g., energy saved, control overhead, scalability, network connectivity, and network performance in terms of network throughput, packet delay, etc., have been considered in the literature.

The metric-based strategies are derived directly from the traditional ad hoc routing schemes by using energy-aware routing metrics instead of distance or delay-based ones. They demonstrate some advantages. For instance, they have few negative impacts on the network performance. Although the paths found by metric-based routing may not be the shortest paths, the paths can be

established immediately and the extra delay incurred is small compared to the sleep-based routing. In addition, alternative paths exist in the network in the same fashion as in traditional routing. Therefore, no network partition problem is involved. This also preserves the original network capacity, in terms of the throughput the network can provide. Furthermore, the control overhead incurred is limited. The energy-aware metrics, such as energy consumption and remaining battery, are easy to attain. Therefore, it's easy for them to integrate with current ad hoc routing protocols and scale to large networks. However, the energy saved by the metric-based routing is very limited, and the improvement on node and network lifetime is not significant. This is due to the fact that all nodes in the network are required to be awake and the idle mode consumes almost as same energy as transmit and receive mode, as indicated in Section 2.3.2.

To conserve more energy, the sleep-based strategies take advantage of the sleep mode, which has very low energy consumption. However, the sleep scheduling strategy has to be created to help the routing protocols avoid using the sleeping nodes while maintaining the network performance. Previously proposed strategies have different characteristics and apply to different scenarios, as discussed in Section 2.3.2. For example, the cluster-based schemes can easily schedule the node status while maintaining the network connectivity, hence avoid the possible significant performance degrading. However, clustering algorithms generate extra overhead in terms of control packets and computing complexity, especially in the case of mobility. Therefore, the cluster-based schemes apply to small or middle size networks consisting of high-capacity nodes, e.g., laptops. The flat schemes remove the clustering overhead, but incur other types of overhead. The dominating set scheme, e.g., Span [8], requires control messages to exchange status and membership information. The location-based scheme, e.g., GAF [61], and the dual radio system, e.g., STEM [53], need extra hardware, and hence increase the network cost. The on-demand scheme [64] incurs high complexity to understand the routing messages and sacrifices network performance due to the network partition to conserve more energy.

In conclusion, sleep-based energy conservation schemes can save more energy, thus more efficiently prolong the network lifetime. However, none of the existing strategies is scalable enough to apply to a large network consisting of thousands of nodes or even more, and simple enough to be integrated into a very low-cost node with extremely limited resources. In the next chapter, a novel sleep scheduling strategy is proposed for this scenario, which incurs extremely low overhead while

maintaining the network connectivity. In Chapter 3, the preliminary work including the description of the strategy and the performance evaluation via analytical and simulation study is presented.

2.4 SUMMARY

Conventional ad hoc network routing protocols, as introduced in the previous sections, require all the nodes keep listening even if there is no traffic or the neighbor nodes are redundant for each other. This wastes a lot of energy and significantly reduces the lifetime of the nodes as well as that of the network. Metric-based energy efficient routing protocols do not reduce the energy consumed by the idle nodes, which do not contribute to the routing. The sleep-based energy efficient routing protocols in the literature prolong the network lifetime but introduce significant overhead and high complexity. In this dissertation, we propose a strategy to improve the energy efficiency of ad hoc and sensor network routing by a simpler way of employing sleep mode. Our design has been driven by the following three goals:

- Simplicity: wireless nodes may have very limited computing capability and memory resources. Therefore minimal operation and information maintenance are desirable. For instance, the state-of-art technology allows a bluetooth radio system to be less than US$10, the price of a pico node is targeted to be less than US$1 and the cost of a sensor node should be much less than US$1 in order for a sensor network to be feasible [2]. Besides the low cost, some of the systems have to be of small size, e.g., smaller than even a cubic centimeter [46]. These stringent constraints lead to the very limited hardware capability.

- Scalability: an ad hoc network can be composed of a large number of nodes. An extreme case for a sensor network may be on the order of millions [2]. In such a large network, the overhead generated by the routing should be kept as low as possible.

- Connectivity: a high degree of the network connectivity enables low path setup delay and quick network response for the application. We try to design energy efficient protocols that do not adversely affect the connectivity and delay sensitive applications.

In addition to these three major design goals, the new strategy should also be adaptive to various network environments. For example, heterogeneous nodes in a network can result in non-uniform node density and varying battery charge. Therefore, an adaptive strategy should be able to tune the major parameters automatically to improve energy efficiency without significantly affecting the performance.

3.0 GOSSIP-BASED SLEEP PROTOCOL

In the previous chapter, we have reviewed the prior work in the area of conventional and energy efficient ad hoc network (including sensor network) routing. Although a great deal of research has been done and many schemes have been proposed to address the energy conservation problem, there is still plenty of room for improvements. In this chapter, we will propose a new energy conservation strategy with extremely low overhead, which is beneficial to applications that require a very low network cost, with extremely constrained resources (e.g., CPU, memory, etc.).

The chapter is composed of the following sections. We give a brief overview of percolation theory in Section 3.1 and gossiping protocols for broadcasting in ad hoc networks in Section 3.2. These two sections are the background our work is based on. In Section 3.3, we propose our gossip-based sleep management protocol. First, a synchronized version for static networks is introduced. Then we argue that it works as well for a network with uniformly distributed mobility. Furthermore, we modify the algorithm and extend it to an asynchronous network. We study the network connectivity problem in Section 3.4 by simulation. Analytical results for the algorithm's energy conservation are presented in Section 3.5. Section 3.6 concludes the chapter.

3.1 BACKGROUND ON PERCOLATION THEORY

The proposed strategy is based on the concepts from percolation theory which we briefly review here [39] [18]. Following the modeling in [18], if we write d ($d \geq 2$) for the dimension of the process and $\mathbb{Z} = \{\ldots, -1, 0, 1, \ldots\}$ for the set of all integers, $\mathbb{Z}^d$ is the set of all vectors $x = (x_1, x_2, \ldots, x_s)$ with integral coordinates and $\delta(x, y) = \sum_{i=1}^{d} |x_i - y_i|$ is the distance from x to y. We may turn $\mathbb{Z}^d$ into a graph, called a *d-dimensional cubic lattice*, by adding edges between

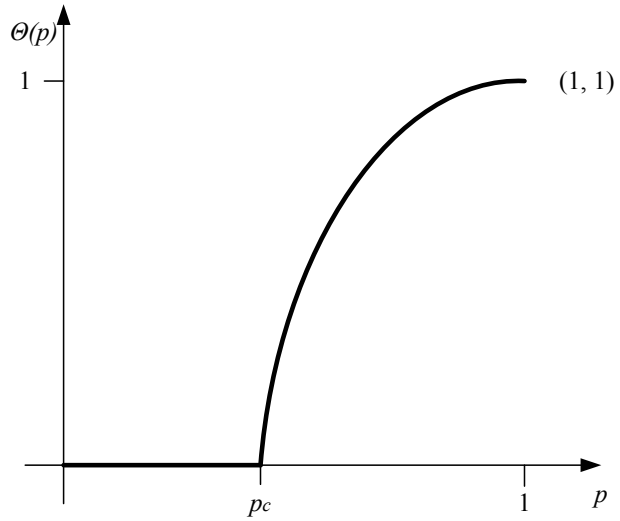

Figure 2: Percolation probability

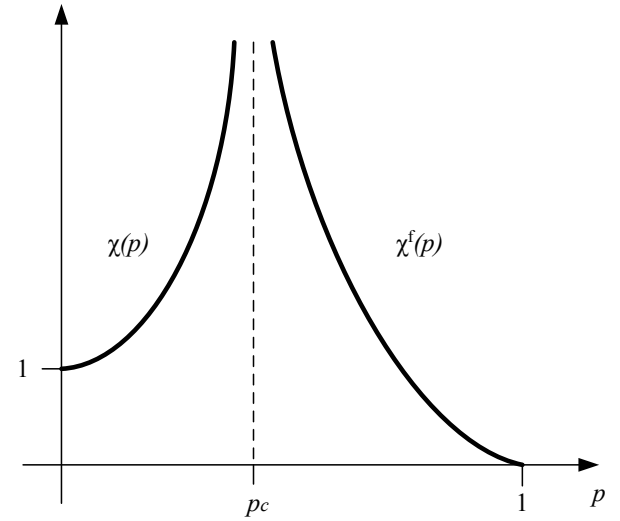

Figure 3: The left-hand curve is a sketch of the mean cluster size $\chi(p)$. The right-hand curve is a sketch of the mean size $\chi^f(p)$ of a finite open cluster when $p > p_c$.

all pairs x, y of points of $\mathbb{Z}^d$ with $\delta(x, y) = 1$. We denote this lattice by $\mathbb{L}^d$, and we write $\mathbb{Z}^d$ for the set of vertices of $\mathbb{L}^d$, and $\mathbb{E}^d$ for the set of its edges.

Let p and q satisfy $0 \leq p \leq 1$ and $p + q = 1$. There are two types of percolation processes: *bond* percolation and *site* percolation. For bond percolation, we declare each edge of $\mathbb{L}^d$ to be *open* with same probability p and *closed* otherwise, independently of all other edges. Similarly, for site percolation, we declare each vertex of $\mathbb{L}^d$ to be *open* with same probability p and *closed* otherwise, independently of all other vertices. In both cases, the lattice $\mathbb{L}^d$ will be grouped into open clusters. We are interested in the size and the shape of the clusters as p varies from 0 to 1. A sketch of the probability that a given node belongs to an open infinite cluster, termed *percolation probability* $\theta(p)$, is shown in Fig. 2 [18]. We can see that there is a threshold p_c, below which there is no infinite open cluster.

The result from percolation theory shows that there exists a critical value $p_c > 0$ such that in the *subcritical phase* (when $p < p_c$), nodes form finite clusters almost surely; in the *supercritical phase* (when $p > p_c$), however, there exist a unique infinite cluster almost surely. More formally (Theorem 1.11 in [18]):

The probability $\psi(p)$ that there exists an infinite open cluster satisfies

$$\psi(p) = \begin{cases} 0 & \text{if } \theta(p) = 0, \\ 1 & \text{if } \theta(p) > 0. \end{cases}$$

The fraction of nodes belonging to an infinite cluster determines the quality of the connectivity. To date, there is unfortunately no explicit expression of this fraction, nor of p_c. A sketch of the mean cluster size when $p < p_c$ and the mean size of a finite open cluster when $p > p_c$ is shown in Fig. 3. Although there is no explicit expression, we can obtain approximations via simulation, as shown in [22].

3.2 GOSSIP-BASED AD HOC ROUTING

In ad hoc networks, gossiping protocols [22] have recently been proposed to reduce the flooding overhead in ad hoc routing protocols, as many routing protocols use some kind of flooding scheme

to send routing messages. With flooding, every node needs to forward the message once, but this is not always necessary since a node with more than one neighbor receives multiple copies of that message. Gossiping reduces this by making some of the nodes discard the message instead of forwarding it. Essentially, a node tosses a coin to decide whether or not to forward the message. The probability p that a node forwards a message is called the gossip probability. Haas, et al., [22] shows that, given a sufficiently large network and a gossip probability p greater than certain threshold, almost all the nodes in the network can receive the message. For example, in a 20×50 grid topology, a value of 0.72 with the first 4 hops from the source node forwarding the message with probability 1 allows almost all the nodes to get the message in almost all the executions of the simulation. This reduces the flooding overhead by about $100 \times (1 - p)\% = 28\%$.

3.3 GOSSIP-BASED SLEEP PROTOCOL (GSP)

Inspired by the results of percolation theory and gossip-based ad hoc routing, we propose the Gossip-based Sleep Protocol (GSP) to achieve energy efficiency in wireless ad hoc networks without the complexity and overhead incurred by other strategies. Our observation is that if gossiping can make all the nodes receive a message, then the nodes forwarding the message are connected at least by the paths the message passes through. Therefore, in a static network without mobility (e.g., a sensor network), if gossiping protocols with certain probability p' [22] can make almost all nodes in the network receive the message, then if all nodes go to sleep with probability $p = (1 - p')$, almost all the awake nodes stay connected. Thus, we can safely put a percentage (p) of the nodes in sleep mode without losing network connectivity. We term p the *gossip sleep probability*.

We assume the network is synchronized, i.e., every node decides its own mode for the next period at the same time. The length of the period T is predefined and we term it the *gossip period*. Basically, every node switches on or off based on probability p, as shown in Fig. 4(a). The basic version of GSP is described as follows and we term it GSP1.

Algorithm 1 (GSP1):

- At the beginning of a gossip period, each node chooses either going to sleep with probability p or staying awake with probability $1 - p$ for this period

- All sleeping nodes wake up at the end of each period
- All nodes repeat the above process for every period

The nodes can be synchronized by a control message at the beginning of each period. For example, we can take advantage of the periodic or event-driven ADV(advertisement) message broadcast from the sink node in Field based Optimal Forwarding [62] without incurring extra overhead. The nodes can also wake up a little bit earlier before the end of each period to wait for the control message and the network performance will not be affected by the extra awake nodes, who are doing nothing but waiting during that short time.

Fairness requires that the length of the period in GSP must be much smaller than the lifetime of the nodes in the network to prevent the condition where a different group of nodes dies in each subsequent period. On the other hand, a longer gossip period avoids frequent link failures.

GSP1 applies to a network without mobility, such as a sensor network. In Haas's work [22] studying gossip based routing, two types of network topologies were studied, namely, regular grid networks and random networks. In this work, our simulation study focuses on the random networks since they are more practical and our analytical study focuses on the grid networks due to their simplicity. Furthermore, we want to know if GSP1 applies to a mobile ad hoc network. A random static network can be constructed by placing nodes randomly in a certain area. In an ad hoc network, random mobility can be viewed as such random placement. At any given instance of time, an ad hoc network with random mobility is a random static network or topology, in which GSP1 can be applied. During a period of time, multiple such topologies are generated. So we can see that although a mobile ad hoc network continuously changes its topology, GSP1 works for the entire lifetime of the network given a random mobility model. Therefore we can extend the GSP1 to the mobile ad hoc networks. Specifically, in an ad hoc network with random mobility, there is a threshold p and if every node goes to sleep for a predefined period with a probability smaller than p, almost all the awake nodes stay connected.

GSP1 requires all nodes in the network synchronized so that they can toss the coin at the same time. With high mobility, this may either be unachievable or incur overhead and complexity. Since we aim at the applications that require low complexity, a simpler protocol without synchronization is always desirable. To remove synchronization and apply to heterogeneous networks, we assume that every node chooses a uniformly distributed random time interval, termed the *gossip interval*,

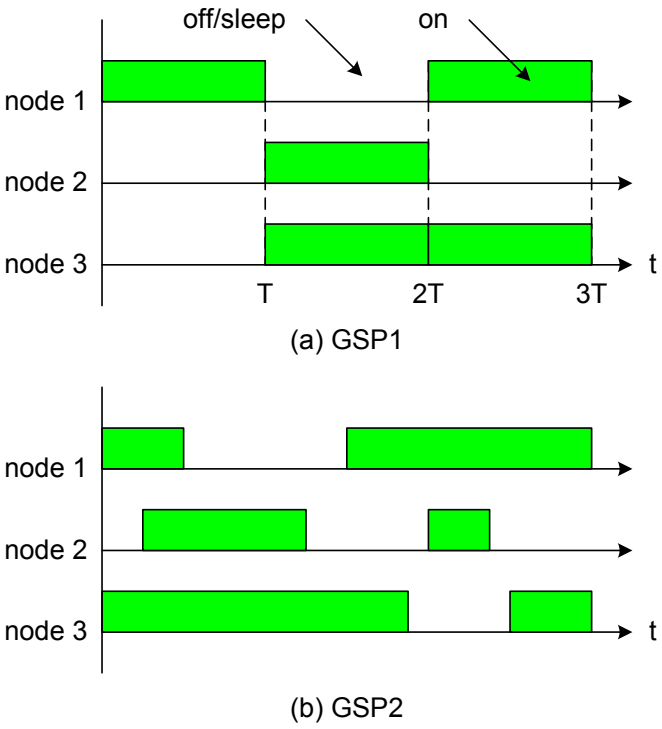

Figure 4: Nodes switch on and off in GSP

independently and after the time is up, the node will choose another random interval immediately. The nodes in the network do not toss the coin at the same time and none of the nodes has any knowledge of other nodes' coin tossing timing. Of course, to make it feasible, we assume the possible maximum gossip interval is much smaller than the lifetime of the network. This process is shown in Fig. 4(b). Now, we have two orthogonal dimensions of randomness, space and time. Obviously, at any time instance, the combination of these two kinds of randomness still results in a random topology to which GSP1 can be applied. In other words, in an ad hoc network with random mobility, there is a threshold p and if every node goes to sleep for an independent random interval with a probability smaller than p, almost all the awake nodes stay connected. So we have the asynchronous version of GSP as follows and term it GSP2.

Algorithm 2 (GSP2):

- Each node independently generates a random time interval and chooses either going to sleep with probability p or staying awake with probability $1 - p$ for the interval.
- Every sleeping node wakes up at the end of its own interval
- Every node repeats the above process for every random interval independently

Unlike other protocols using a sleep mode (e.g., cluster-based schemes, SPAN and GAF), GSP is extremely simple and requires almost no information, even from immediate neighbors. The gossip sleep probability p is purely dependent on the network density and can be configured before the deployment of the network. GSP improves upon the energy consumption by schemes such as Span and GAF by not requiring nodes to transmit and receive additional network maintenance traffic. On the other hand, GSP is expected to provide less improvement of the network lifetime than other schemes due to the limited knowledge of the network, which contributes to the simplicity as we just mentioned. Therefore, GSP is more suitable to the large low-cost network, which desires low complexity to reduce the cost of every node as much as possible.

The major objective of GSP is to achieve energy efficiency by putting some nodes in a sleep mode. The potential disadvantage of this approach is that packets may go through longer paths if the nodes sleeping are on the shortest paths between source and destination nodes, resulting in more energy consumption in the network-wide communication. Also, paths will be broken more often due to the mode change of the nodes. Therefore, more overhead is generated to overcome

the path failures and this will consume some extra energy. So an open issue is does the energy saved by GSP exceed the extra energy consumed by non-optimal paths and extra routing overhead. In addition, path failures due to sleeping may decrease the network throughput and increase the end-to-end delay. In order to evaluate these tradeoffs with GSP, we have conducted a discrete event simulation based performance study. Before that, an analytical study for grid topologies is presented, in which another set of simulation is used to study the change of the path length with GSP.

3.4 CONNECTIVITY STUDY

Maintaining the connectivity of the network is one of our design goals. One of the basic methods to study this performance is to conduct a connectivity measurement on a network with and without GSP.

3.4.1 Simulation Model

To isolate the effects of all factors other than GSP, there should be no protocols running or traffic transmitting in the network except GSP. Therefore, the network is basically a graph. There are 200 nodes in the graph and they are uniformly randomly placed within a $2000m \times 2000m$ area. We assume every node's transmission range is $250m$ and two nodes are directly connected if their distance is less than $250m$.

In addition, we want our scenarios close to a real network, so we make the nodes in the graph move randomly following the Random Waypoint mobility model [30]. In Random Waypoint, each node begins at a random position, picks a new random position to which to move, and moves there in a straight line at a random speed. Each node independently repeats this behavior and the average degree of mobility is varied by making each node remain stationary for a period called the pause time every time before it moves to the next position. The smaller the pause time, the higher the average mobility. In our simulation, the maximum speed of the nodes is 20 m/s and the pause time is 0, i.e., the highest level of mobility is studied.

The total simulation time for one run is 100 seconds and we average 20 runs for each case. The asynchronous GSP (i.e., GSP2) is studied with the gossip interval set to 10 seconds. We vary the gossip sleep probability(p) from 0 to 0.9 to measure the connectivity of the graph. We observe the graph every second, hence, we got 20 data points for one run and 2000 points for 20 runs of one case. The final results are the average of these 2000 data points.

3.4.2 Results

The performance metrics we use to measure the graph connectivity are the number of nodes in the largest connected component(LCC) in the graph and the coverage ratio R_c. Let's designate N_a the total awake nodes in the graph. The coverage ratio is defined as follows:

$$R_c = \frac{LCC}{N_a} \tag{3.1}$$

If the awake nodes are uniformly distributed in the area, the coverage ratio R_c can be used to roughly estimate the area covered by the LCC, since the awake nodes are supposed to distribute uniformly. Of course, R_c is not very accurate in certain extreme cases. For example, in a large network with only two nodes awake, the most likely $N_c = LCC/N_a = 50\%$ doesn't mean 50% of the area is covered. However, the value of R_c is usually low and unacceptable in such extreme cases. For instance, the above 50% is not a good value in our study. Another extreme case is only one node awake and $R_c = 100\%$. We can easily recognize that this result is not very useful when considered together with LCC. Therefore, we'd say R_c just roughly estimates the coverage.

The results are shown in Table 2, where, D_a is the average node degree (i.e., number of neighbors) of the awake nodes. We can see from the table that most of the awake nodes are connected very well if the gossip sleep probability $p < 0.5$, i.e., more than 90% awake nodes are connected. This result is compatible with percolation theory and our analysis for GSP. Based on this result, we can see the basic approach of GSP and start to study its various performance in a real network, as shown in the rest of this work.

Table 2: Connectivity Measurement of a Graph with Random Mobility

p	D_a	N_a	LCC	R_c
0.0	12.18	200.00	198.94	0.9947
0.1	11.03	179.85	177.90	0.9892
0.2	9.96	159.39	156.20	0.9799
0.3	8.50	140.05	136.89	0.9774
0.4	7.20	120.26	114.08	0.9479
0.5	6.10	100.02	90.88	0.9061
0.6	4.81	80.38	65.18	0.8073
0.7	3.63	60.81	38.61	0.6296
0.8	2.39	39.62	16.30	0.4042
0.9	1.13	19.74	5.37	0.2696

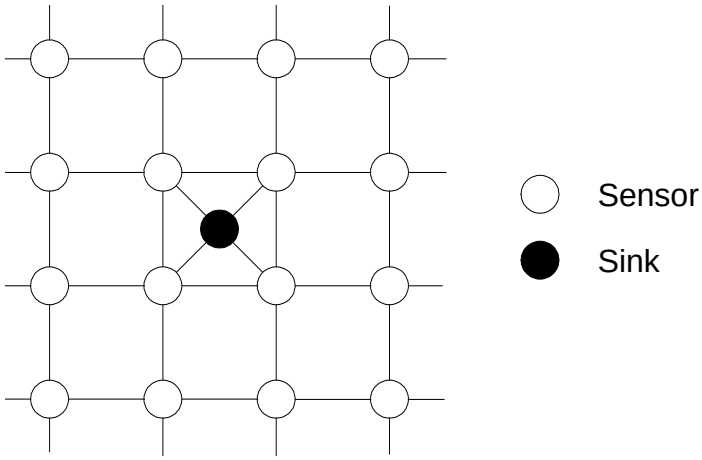

Figure 5: The central area of the grid topology used by the simulation.

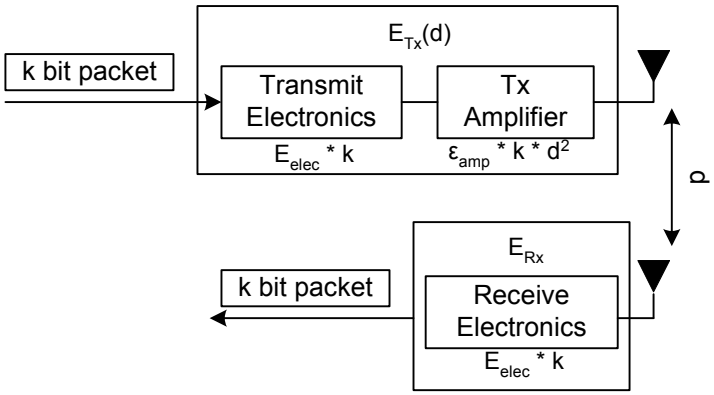

Figure 6: Radio model.

3.5 PERFORMANCE ANALYSIS

In the following analysis, we only consider static grid topologies with a single data sink node as illustrated in Fig. 5. In this case, the network is more like a static sensor network and we will introduce some parameters of sensor networks into the analysis. We assume that all calculations are based on the period of time to transmit one bit of data, i.e. *bit time*. Transmissions are actually a frame which we will discuss in section 3.5.6. We also assume the traffic load remains constant with or without GSP, i.e., the number of bits generated by the sensors in a bit time are the same. Although the actual application may generate bursty traffic, this assumption will not change our results in that the energy consumption incurred is based on the amount of the traffic, not the distribution of the traffic.

3.5.1 Radio Model

We adopt the radio model given in [24], as shown in Fig. 6, and follow their notation in our analysis and simulations. Specifically, the radio dissipates $E_{elec} = 50nJ/bit$ to run the transmitter or receiver circuitry and $\epsilon_{amp} = 100pJ/bit/m^2$ for the transmit amplifier to achieve an acceptable signal-to-noise ratio. As in [24], we also assume an r^2 path loss model to describe the energy loss due to channel transmission. Although many other radio models and path loss models exist, we

Table 3: Radio Characteristics

Radio mode	Energy Consumption
Transmit($E_{elec} + \epsilon_{amp}$)	$50nJ/bit + 100pJ/bit/m^2$
Receive(E_{elec})	$50nJ/bit$
Idle(E_{idle})	$40nJ/bit$
Sleep	0

expect they will not change our analytic results but only the amount of final energy conserved or workable scenarios (e.g. traffic load, network size). Additionally, we assume that an idle receiver consumes $E_{idle} = 40nJ$ in the period of transmitting or receiving a bit. The difference between this value and the energy consumption of receive mode is relatively larger than the existing sensors [47] and this will decrease the performance of GSP. Thus, for simplicity, we assume a node sleeping doesn't dissipate any energy, although in reality it's a small value close to zero. The above radio characteristics are summarized in Table 3.

3.5.2 GSP Performance Analysis

In the remainder of this section and the next section, we study how much energy can be saved by employing GSP in the network. By randomly applying sleep mode to some nodes, GSP may not be able to establish the optimal path between two nodes if some of the nodes on the path are in sleep. To achieve energy efficiency, GSP must conserve more energy by employing sleep mode than is consumed by the longer average path length incurred. If we use L_{GSP} and L_{min} to represent the average path length with and without GSP respectively, N is the number of nodes in the network, the average total energy consumption $E_{non-GSP}$ during a bit time without GSP can be calculated by Equation 3.2.

$$E_{non-GSP} = (E_{elec} + d^2 \times \epsilon_{amp}) \times B \times L_{min} + E_{idle} \times (N - (B \times L_{min})) \tag{3.2}$$

where, B is the traffic load, i.e. the number of bits generated during a bit time in the entire network. d is the distance between nodes.

Here we assume every traffic source transmits as fast as possible to keep all the intermediate nodes busy. The first term of Equation 3.2 is the transmission energy consumed by all the nodes in the network that have traffic to send. The second term is the energy consumed by the rest of the nodes. Although some of them are in receive mode, for simplicity, we assume all of them are in idle mode. This assumption makes us underestimate the energy consumed by the protocols without GSP, thus underestimate the performance improvement of GSP.

Similarly, the average total energy consumption during a bit time with GSP can be calculated by Equation 3.3. The difference is the second term, since the total number of the idle nodes is reduced.

$$\begin{aligned} E_{GSP} &= (E_{elec} + d^2 \times \epsilon_{amp}) \times B \times L_{GSP} \\ &\quad + E_{idle} \times (N \times (1-p) - (B \times L_{GSP})) \end{aligned} \tag{3.3}$$

If we define E_{diff} as the difference between $E_{non-GSP}$ and E_{GSP}, we get Equation 3.4. If E_{diff} can be greater than zero then GSP can reduce the energy consumption of the network.

$$\begin{aligned} E_{diff} &= E_{non-GSP} - E_{GSP} \\ &= E_{idle} \times N \times p - (E_{elec} + d^2 \times \epsilon_{amp} - E_{idle}) \times B \times (L_{GSP} - L_{min}) \end{aligned} \tag{3.4}$$

For clarity, we define α as the ratio of average extra path length with GSP, i.e.

$$\alpha = (L_{GSP} - L_{min})/L_{min} \tag{3.5}$$

So Equation 3.4 can be written as follows

$$E_{diff} = E_{idle} \times N \times p - (E_{elec} + d^2 \times \epsilon_{amp} - E_{idle}) \times B \times L_{min} \times \alpha \tag{3.6}$$

The first term of Equation 3.6 is the energy saved by GSP due to the sleep mode, and the second term is the extra energy consumed by GSP due to the longer average path. $B \times L_{min}$ is the total bits in the network at any given time and $B \times L_{min} \times \alpha$ is the extra number of bits in the network since data must travel through a longer path. From Equation 3.6, we know that more energy can be saved when a network has larger number of nodes N and higher gossip sleep probability p. However, a high sleep probability can lead to a partitioned network. Obviously, the optimal value of p is dependent on different network scenarios. Also, one can observe that the extra

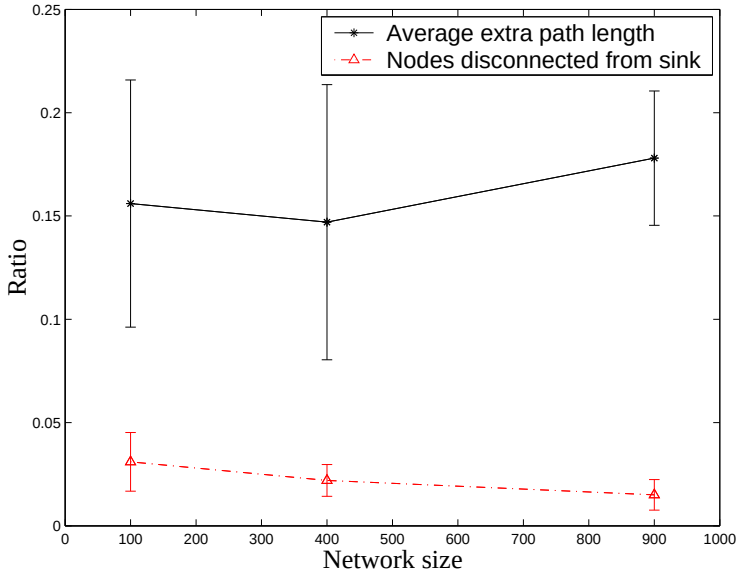

Figure 7: The ratio of average extra path length (α) and the nodes disconnected from the sink when p=0.25, with a 90% c.i.

energy consumption will be increased when the network has higher traffic load and longer average path. The sensor node number N and the traffic load B depend on the specific network scenario and the application. L_{min} and α are also dependent on the sensor network scenario. However, for a grid topology, L_{min} is fixed for a given network, we just need to study α with respect to various topologies and values of p.

3.5.3 Simulation Model for α

We utilized the ns-2 network simulator [13], with the CMU Monarch Project wireless and mobile ns-2 extensions, to study the effects of employing GSP. To study the change of average path length with network size, three grid topologies with a single sink node in the center are used, 10×10, 20×20 and 30×30. The central area of the topologies is shown in Fig. 5, and there are a total of 101, 401 and 901 nodes respectively, i.e. 100, 400 and 900 sensors and a sink. Each node has four immediate neighbors. We assume that the sink is not power limited. In the simulation, all nodes are awake without GSP and $(1-p)\%$ of nodes are awake when GSP is utilized. We use the Destination-

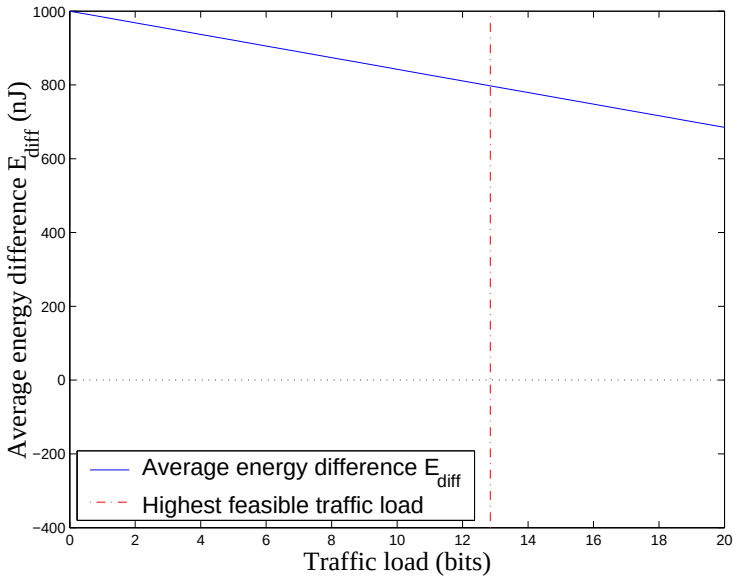

Figure 8: Energy difference (E_{diff}) between $E_{GSP-saved}$ and $E_{GSP-extra}$ vs. traffic load (B) in bits when $p = 0.25$, $N = 100$, $L_{min} = 5.05$ and $\alpha = 0.156$.

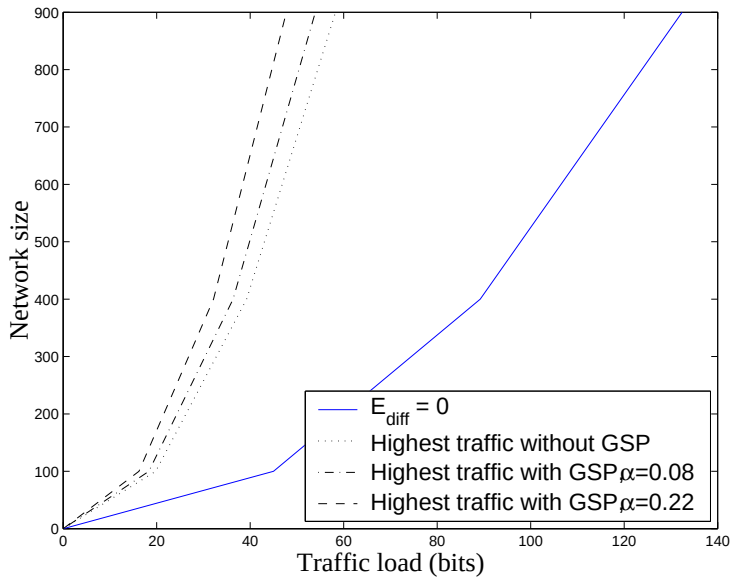

Figure 9: Number of nodes N vs. traffic load B (bits) when $p = 0.25$, $\alpha = 0.22$ and $E_{diff} = 0$.

Sequenced Distance Vector (DSDV) protocol [43], a proactive ad hoc routing protocol, to find the length of the shortest path from every sensor to the sink. DSDV is a convenient tool to study the average path. However, GSP does not require its use. Since the MAC protocol doesn't affect the path length as long as it provides routing protocols correct information on neighboring nodes, we use IEEE 802.11b as the MAC layer protocol. The gossip sleep probability in the simulation is 0.25, i.e. p=0.25, approximately the highest value resulting in a connected network (Fig. 7). The simulation results are the average of 20 runs with a 90% confidence interval.

3.5.4 Simulation Results for α

Fig. 7 shows the simulation results of α as a function of N. As expected from the discussion above, the average path length with GSP is longer than without GSP. For example, without GSP, the average path length is 5.05 for the 10×10 grid topology. With GSP, the value of this variable is 5.838, with a 90% confidence interval (5.537, 6.140). The result shows that the average path length increases by 15.6%, i.e. α=(5.838-5.05)/5.05≈0.156. Fig. 7 shows that α does not vary much over the tested topologies.

In addition, we found some nodes are disconnected from the sink with GSP. For example, in the 10×10 grid topology there are 3.11 sensors on average are separated from the sink with a 90% confidence interval (1.684, 4.527). The average ratio is 3.11/100=0.0311. As the network grows larger, we can see the trend that the ratio of the disconnected nodes decreases.

3.5.5 Continued Performance Analysis

Using the simulation results of Fig. 7 in Equation 3.6 we see the possible energy savings of GSP. To get a more visualized result, we assume that each node is 10 meters apart from one another. Fig. 8 shows the value of E_{diff} for the 10×10 grid topology (solid line) with respect to the traffic load B. With around 25% of nodes in sleep the feasible highest traffic being transmitted in the entire network during a bit time is only about 75 bits, so the feasible $B \leq 75/L_{GSP} = 75/(L_{min} \times (1+\alpha)) = 12.8$. At this point and in the area smaller than it, E_{diff} is positive. It is worth noting that although the area of $12.8 \leq B \leq 100/L_{min} = 19.8$ is feasible to non-GSP protocols, it's not feasible to GSP. In other words, we should not employ GSP or we should use

a smaller p when the traffic is this high. Here we consider the worst case only. If we assume the perfect MAC layer protocol and a node can not transmit and receive data at the same time, the feasible highest traffic load for the above two cases is only about 6.4 bits and more energy can be saved.

Fig. 9 shows the situations in which GSP can be employed. It is a plot of network size N vs. traffic load B when the gossip sleep probability is 0.25. We obtain the solid curve by making Equation 3.6 equal to zero and assuming the ratio of average extra path cost (α) is always 0.22 for different network size, which is the worst case in our simulation as shown in Fig. 7. The area above this curve represents the positive energy difference (E_{diff}) that leads to energy savings when using GSP. The dotted curve is the feasible highest traffic load without GSP, i.e., N/L_{min}. The dashdot and dashed curves represent the feasible highest traffic load with GSP, i.e., $N/(L_{min} \times (1+\alpha))$, but with the lower bound and upper bound of α, i.e., 0.08 and 0.22, respectively. The areas above these curves are feasible.

3.5.6 Analysis at Frame Level

The above analysis is based on the level of bit time. In practice, data is transmitted in frames. In this subsection, we extend our analysis to frames and define the time to transmit a frame *frame time*. At the frame level, we re-write Equation 3.2, 3.3 and 3.6 as follows.

$$\begin{aligned} E'_{non-GSP} &= (E_{elec} + d^2 \times \epsilon_{amp}) \times F \times L_{min} \times S \\ &\quad + E_{idle} \times (N - F \times L_{min}) \times S \end{aligned} \tag{3.7}$$

where, F is the traffic load in frames, i.e. the number of frames generated during a frame time in the entire network. S is the average number of bits in a frame.

$$\begin{aligned} E'_{GSP} &= (E_{elec} + d^2 \times \epsilon_{amp}) \times F \times L_{GSP} \times S + E_{idle} \\ &\quad \times (N \times (1-p) - F \times L_{GSP}) \times S \end{aligned} \tag{3.8}$$

and

$$\begin{aligned} E'_{diff} &= E'_{non-GSP} - E'_{GSP} \\ &= [E_{idle} \times N \times p - (E_{elec} + d^2 \times \epsilon_{amp} - E_{idle}) \times F \times L_{min} \times \alpha] \times S \end{aligned} \tag{3.9}$$

From Equation 3.9, we can easily see that E'_{diff} is similar to E_{diff} if S is a constant, i.e. the frame size is fixed. In one frame time, the number of frames being transmitted in the entire network (F) cannot be larger than the number of nodes, which is same as traffic load B at the bit level. So Fig. 8 and 9 also apply to E'_{diff} except the traffic load is F in term of frames.

3.6 SUMMARY

In this chapter, we propose a new energy conservation strategy for ad hoc network routing, i.e., Gossip-based Sleep Protocol(GSP). The core idea is to employ probabilistic based sleep modes - essentially, tossing a coin to decide whether or not a node should sleep for the next period. The strategy is very suitable to a large and low cost network. This basic idea is intuitively proposed for a synchronous network without mobility, e.g., a wireless sensor network. Then we show that it also applies to a mobile network. Furthermore, to remove the synchronization overhead, an asynchronous version of GSP is proposed. By introducing a sleep mode into the network, the total energy consumption of the network can be reduced and the network lifetime can be prolonged. However, the sleep mode may increase the length. Therefore, analysis is conducted to study the effect of GSP on the network lifetime.

From the preliminary study, we can see that we achieved our design goals. We achieved the simplicity by only adding a local timer to each node. Once its time is up, every node decides whether to go to sleep in the next gossip period/interval with the gossip sleep probability p. The property of gossiping makes it scalable to very large networks. Network connectivity is decided by the gossip sleep probability p. We show that with certain value of gossip sleep probability p and under certain topology density, the network is still connected. In the next chapter, we will conduct simulation to show that the performance of the network is only slightly affected by GSP.

Similar to the protocols introduced in Chapter 2, in our work, we introduce the sleep mode concept into conventional ad hoc routing protocols to trade off network density for energy efficiency. However, compared with other techniques, the proposed GSP is very simple and scalable without maintaining any information except a local timer. Our scheme is totally flat and other flat or cluster-based protocols can be used over our scheme to further reduce energy consumption. For

example, the awake nodes in our protocol can be grouped into clusters and thus utilize their energy more efficiently. Metric-based energy efficient routing schemes can also be integrated with our protocol to find an energy efficient path from the resulted topology.

4.0 PERFORMANCE EVALUATION

This chapter presents a comprehensive simulation study on GSP. Firstly, we present in Section 4.1 simulation results to show the correctness and effectiveness of GSP. We also compare the results with the analytical results from Chapter 3 to justify our analysis and simulation methodologies. Secondly, a more detailed simulation study is presented in Section 4.2. We study the performance of GSP in different scenarios by changing various parameters. Then in Section 4.3, we present some heuristics as general design rules of GSP based on the simulation results. Finally, we discuss the impacts of different ad hoc routing protocols when we integrate them with GSP. The chapter is concluded in Section 4.5.

4.1 PRELIMINARY SIMULATION

If we assume that there is no traffic and routing overhead in the network, all awake nodes stay in the idle mode and the sleep mode does not consume any energy, then with gossip sleep probability p the network lifetime should ideally be prolonged by a percentage p. In practice, the improvement will be less than p. One reason is that the sleep mode consumes non-zero energy although it's very small compared to idle mode. More importantly, the longer paths caused by non-optimal routing and the extra routing overhead caused by more frequent path failures will consume extra energy compared to routing without GSP. The value of p depends largely on the density of a network. Therefore, we expect to see a larger improvement in a denser network. Additionally, since GSP is able to maintain the connectivity of the network with a proper value of p, the throughput and packet end-to-end delay are not expected to be affected too much given a light or moderate traffic load. To confirm the above analysis, we developed a simulation to study the performance of GSP

Table 4: Energy Consumption Model for Lucent IEEE 802.11 WaveLAN PC Card with $2Mbps$

Radio mode	Energy Consumption (W)
Transmit	1.327
Receive	0.967
Idle	0.844
Sleep	0.066

in the remainder of this section.

4.1.1 Simulation Model

We use ns-2 network simulator [13], with CMU Monarch Project wireless and mobile ns-2 extensions, to study the characteristics of GSP. The distributed coordination function (DCF) of IEEE 802.11(b) for wireless LANs is used as the MAC layer. The radio model is similar to Lucent's WaveLAN, which is a shared media radio with a nominal bit rate of 2Mb/sec and a nominal radio range of 250 meters.

Note that GSP requires no information from the routing algorithms and can be integrated with a number of routing protocols. Here, we use Dynamic Source Routing (DSR) [29, 30] as an example to describe how GSP works. We attached GSP to DSR to get GSP+DSR.

The simulation results presented in this work are based on scenarios randomly generated by CMU ns-2 extensions. We use 50 and 100 transit nodes to study the density effects and nodes are randomly placed within a $1500m \times 300m$ area. The node mobility model is Random Waypoint [30]. Although this model fails to provide a steady state and the average nodal speed consistently decreases over time [63], we expect it will not affect our simulation results since the average speed decreases slowly and affects little energy consumption. In Random Waypoint, each node begins at a random position, picks a new random position to which to move, and moves there in a straight line at a random speed. Each node independently repeats this behavior and the average degree of mobility is varied by making each node remain stationary for a period called the pause time before it moves to the next position. The smaller the pause time, the higher the average mobility. In our simulation, the maximum speed of the nodes is 20 m/s and the pause time is varied between 0 and

Table 5: GSP Simulation Parameters

Parameters	Values	Parameters	Values
Simulation time	$\geq 400sec$	Bandwidth	$2Mb/s$
Physical layer	IEEE 802.11b	Max. speed	$20m/s$
MAC layer	IEEE 802.11b	Radio range	$250m$
Traffic model	CBR	Gossip period	$20sec$
Packet size	$512bytes$	Traffic nodes	10
Network size	$50/100nodes$	Packet rate	$10pkt/sec$
Area size	$1500m \times 300m$	Pause time	$0 - 1000s$

1000 seconds. Note that when the pause time is 1000 seconds, the network is static. Besides the transit nodes, there are 10 traffic nodes, which are the source and the sink of the traffic. The traffic is unicast between node pairs and there are five node pairs, i.e., five traffic flows in the network. CBR traffic based on UDP is used. Each packet carries 512 bytes of data payload, making the packet size 532 bytes including an IP header. The packet rate is 10 packets/sec.

Our energy consumption model is based on Feeney and Nilsson's measurements of an IEEE 802.11b Lucent WaveLAN wireless network interface operating in an ad hoc networking environment [14]. Their measurements are summarized in Table 4, where we can see the sleep mode consumes only a tiny fraction of the energy of the other modes. Additional measurement values in the literature evaluating other 802.11b vendor equipments show similar energy consumption rates.

To make sure the traffic does not stop before the network dies, we give traffic nodes infinite energy. The transit nodes have enough energy so that the DSR protocol can run for 400 seconds. Since all nodes in DSR keep listening even without traffic, they run out energy almost at the same time. Also, to mitigate the effects of traffic nodes, we make traffic nodes neither run GSP nor forward traffic in DSR. However, traffic nodes do follow the same mobility model as transit nodes and maintain their own connections as required by DSR.

The parameters for GSP are chosen to show the properties of GSP and they are not necessarily the optimal values. We assume synchronization for GSP1 and use a fixed 20 seconds as the gossip period. The gossip sleep probability is varied as shown in the simulation result figures. Each data point is an average of at least ten runs. The simulation parameters are summarized in Table 5.

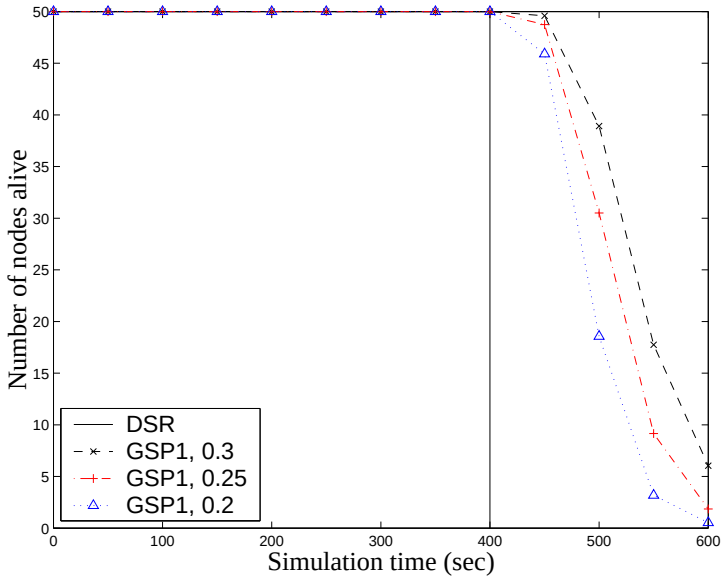

Figure 10: Network lifetime of DSR and GSP1 with different gossip sleep probability (p)

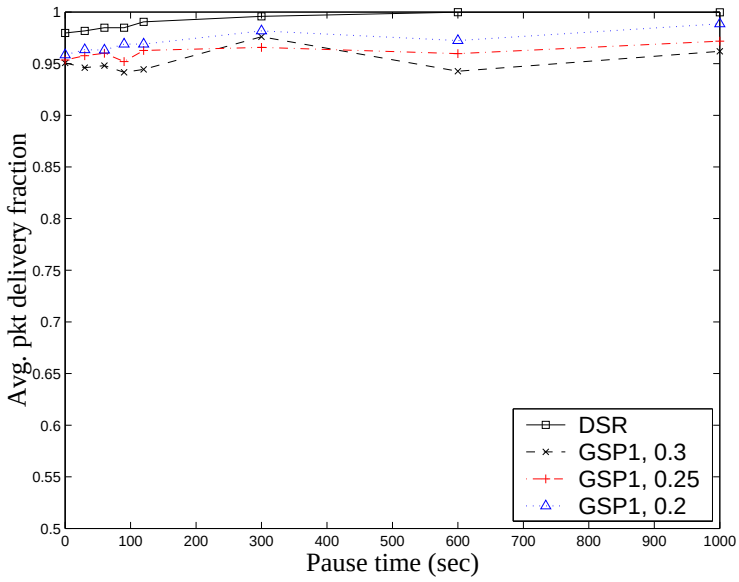

Figure 11: Average packet delivery fraction of DSR and GSP1 with different gossip sleep probability (p)

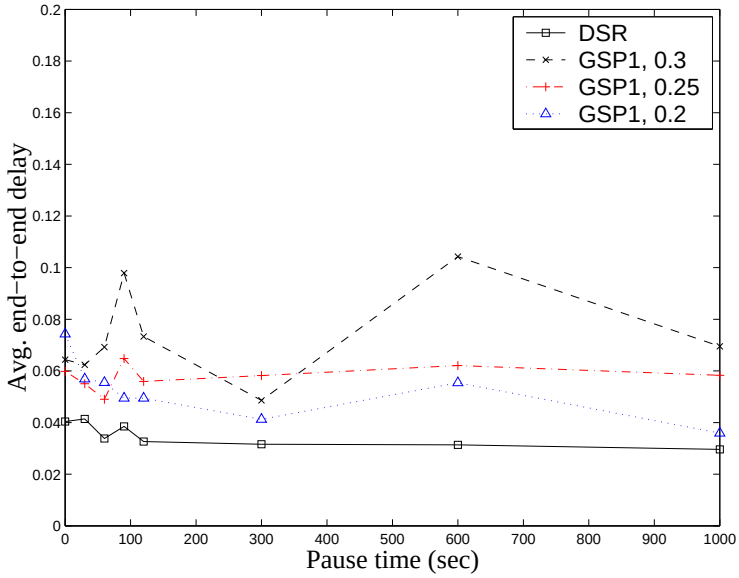

Figure 12: Average end-to-end delay of DSR and GSP1 with different gossip sleep probability (p)

4.1.2 Simulation Results

We evaluated three performance metrics defined as follows:

- Packet delivery fraction: the ratio of the packets delivered to the destination to those generated by the CBR sources;
- End-to-end delay: the delay experienced by each packet, including queuing delays, route discovery delays, retransmission delays at the MAC layer and the salvage process of DSR, and propagation delays;
- Network lifetime: the simulation time that a fraction of transit nodes are alive and the network maintains an acceptable packet delivery fraction and end-to-end delay.

The definition of network lifetime is ambiguous for a given network. Unlike analysis, in practice we may not be able to make all nodes die at the same time to get a definite lifetime value. In addition, we may be more interested in maintaining an acceptable network performance rather than the number of alive nodes in the network. Since the definition of *acceptable network performance* depends on different applications in reality, the value of network lifetime also depends on

the applications. Although the definition is vague, we still can roughly know the lifetime value and clearly see the differences in network performance and proxy metrics such as the percentage of network nodes alive can be used to estimate the lifetime.

To illustrate the basic GSP concept in a MANET, we first compare GSP1 with DSR in a network of 50 transit nodes. Fig. 10 shows the number of nodes alive with respect to the simulation time. GSP1 successfully extends the network lifetime and larger sleep probabilities generate longer extensions. It is worth noting that GSP itself is independent of the mobility pause time. Although not presented here, the simulation results show that, given a sleep probability, the results for different pause times are almost same in our simulation scenarios. So every curve in Fig. 10 is an average of all results for multiple levels of mobility, from static to constant moving.

Fig. 11 and Fig. 12 show the packet delivery fraction and end-to-end delay with respect to the pause time for the first 400 seconds. Note that after this time the DSR network fails. As we expected, with a small gossip sleep probability, GSP does not significantly affect the performance of DSR in terms of average packet delivery fraction. In terms of mean end-to-end delay, as expected, the delay increases with p in GSP as compared to DSR.

To study the effects of network density, we run a similar simulation in the network of 100 transit nodes with pause time 0. With a higher density, a larger sleep probability can be employed and the network connectivity can still be maintained. Again, Fig. 13 shows that the lifetime extension is proportional to the gossip sleep probability and GSP can benefit from the network density. Fig. 14 shows our results of monitoring the packet delivery fraction for every 50 seconds for GSP in the same scenario. We can see, although the average packet delivery fraction of GSP is slightly decreased, it can be maintained for a much longer time.

In both Fig. 13 and Fig. 14, we also show the results of GSP2. The gossip interval in GSP2 is uniformly distributed between 0 and 40 seconds, with an average of 20 seconds. The results show that GSP can work well even without synchronization. Actually, the figures show that GSP2 achieved slightly better performance than GSP1. We attribute this to the fact that higher randomness can smooth and more evenly distribute the power consumption of a network.

We are also interested in the effects of gossip period. A simulation is conducted on the network of 100 transit nodes with pause time 0 and we use GSP1 with gossip sleep probability $p = 0.3$. The gossip period varies from 5 seconds to 100 seconds. The results of network lifetime and average

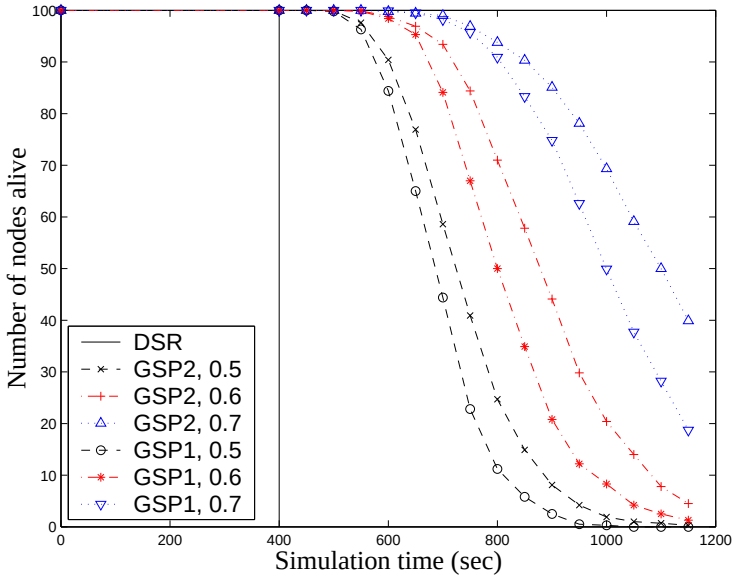

Figure 13: Network lifetime of DSR, GSP1 and GSP2 with different gossip sleep probability (p) in the network of 100 transit nodes with pause time 0

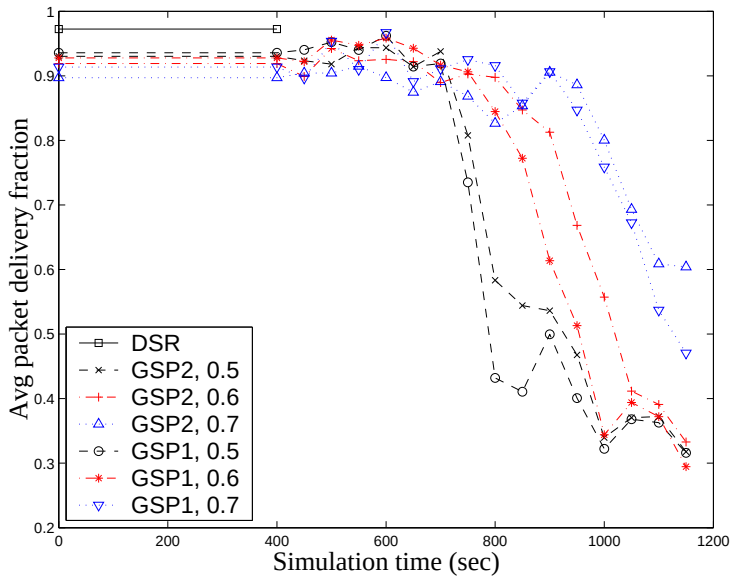

Figure 14: Average packet delivery fraction of DSR, GSP1 and GSP2 with different gossip sleep probability (p) in the network of 100 transit nodes with pause time 0

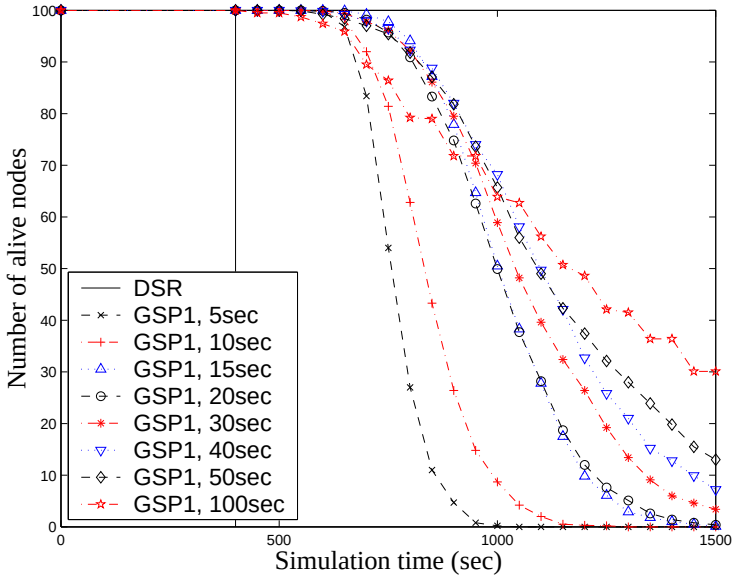

Figure 15: Network lifetime of DSR and GSP1 awith different gossip period in the network of 100 transit nodes with pause time 0, $p = 0.3$

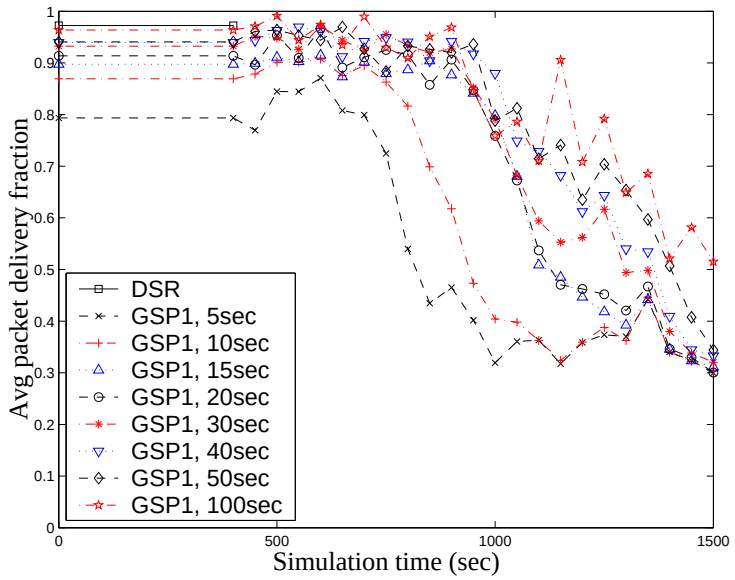

Figure 16: Average packet delivery fraction of DSR and GSP1 with different gossip period in the network of 100 transit nodes with pause time 0, $p = 0.3$

packet delivery fraction are shown in Fig. 15 and 16 respectively. We can see, in general, a larger period extends the network lifetime more and affects the network performance less. Obviously, a smaller period incurs more failures and control overhead to recover these failures. This consumes more energy and drops more packets. From Fig. 15, we also can see that the decay part of the lifetime curve of a smaller period is steeper and smoother. A gossip period much smaller than the lifetime can reduce the overuse of a certain group of nodes and make all nodes consume their energy at the same rate and die at the same time.

4.1.3 Comparison with Analytical Results

In this section, we compare the energy savings measured from the simulation with the ones derived from the analytical model in Section 3.5 to check the correctness of the analysis.

Let $\mathcal{L}$ be the lifetime improvement of GSP, defined as follows,

$$\mathcal{L} = E_{diff}/E_{GSP} \tag{4.1}$$

Let $\mathcal{L}_{GSP}$ and $\mathcal{L}_{non}$ be the lifetime of with and without GSP, we have,

$$\mathcal{L}_{GSP} = \mathcal{L}_{non} \times (\mathcal{L} + 1) \tag{4.2}$$

From Equation 3.4 and 4.1 we get,

$$\begin{aligned} \mathcal{L} &= E_{non-GSP}/E_{GSP} - 1 \\ &= \frac{(E_{elec} + d^2 \times \epsilon_{amp}) \times B \times L_{min} + E_{idle} \times (N - (B \times L_{min}))}{(E_{elec} + d^2 \times \epsilon_{amp}) \times B \times L_{GSP} + E_{idle} \times (N \times (1-p) - (B \times L_{GSP}))} - 1 \\ &= \frac{P_{tx} \times B \times L_{min} + P_{idle} \times (N - (B \times L_{min}))}{P_{tx} \times B \times L_{GSP} + P_{idle} \times (N \times (1-p) - (B \times L_{GSP}))} - 1 \end{aligned} \tag{4.3}$$

where, P_{tx} and P_{idle} are the power consumption of transmission and idle modes respectively.

Table 6 is the simulation results of the average path length with a 90% confidence interval for GSP1 in 50-node networks (average node degree is around 13). The gossip sleep probability is 0.3 and the data is collected for the first 400 seconds for a fair comparison. Similarly, Table 7 shows the results of GSP1 in 100-node networks with gossip sleep probability of 0.5. The case of highest mobility, i.e., pause time 0, is studied.

Table 6: Average Path Length with 90% c.i. of GSP1 in 50-node Networks with Sleep Probability=0.3

Pause time(sec)	DSR(hop)	GSP(hop)
0	2.46(2.23,2.68)	2.42(2.20,2.63)
30	2.61(2.44,2.79)	2.56(2.40,2.72)
60	2.74(2.55,2.94)	2.67(2.45,2.88)
90	2.78(2.46,3.09)	2.79(2.50,3.08)
120	2.89(2.62,3.15)	2.81(2.55,3.06)
300	3.28(2.74,3.83)	3.31(2.76,3.86)
600	3.37(2.98,3.75)	3.28(2.91,3.66)
1000	2.92(2.61,3.23)	2.99(2.65,3.32)
Average	2.88	2.85

Table 7: Average Path Length with 90% c.i. of GSP1 in 100-node Networks with Sleep Probability=0.5

Pause time(sec)	DSR(hop)	GSP(hop)
0	2.40(2.20,2.59)	2.37(2.17,2.57)

From the simulation we can see that the average path lengths of GSP and DSR in both cases are very close. We notice that GSP uses shorter paths sometimes during the simulation and we attribute it to the following reasons:

- There are more collisions in a denser network, therefore, the path request and reply messages via a longer path may go faster. DSR will take the path with the path reply message coming back first.
- Mobility may make paths longer since DSR will not switch to a new shorter path as long as the current one is not broken. But GSP will add a few more path failures and force a little more route refreshing. We can see this from the simulation results of 50-node networks: higher mobility→more failures→shorter paths.

To verify the above analysis, we took a couple of snapshots, shown in Figure 17- 22, of one simulation run in the 50-node network with pause time equal to 0. The figures are generated by the visualization tool NAM built in NS-2. The node location is not presented in real time in case of mobility. Under the Random Waypoint model, nodes in NAM stay in their previous locations until the time instance when they move to the next location, with no movement shown in between. The large black circles in the figures are the animation of radio waves of sending nodes.

- At time 161, DSR is using path 50-44-25-40-55 to route traffic between node 50 and 55, as shown in Figure 17. However, GSP is using 50-8-47-55 with node 44 and 25 in sleep mode (hexagon nodes in the figure), as shown in Figure 18.
- At time 260, DSR is using path 52-15-36-37-57, while GSP is using path 52-9-7-57, which are shown in Figure 19 and 20 respectively.
- At time 300, DSR is using path 54-7-38-11-39-59, while GSP is using 54-44-6-42-59 with node 7 and 38 in sleep, which are shown in Figure 21 and 22.

This phenomenon doesn't conflict with the one in Section 3.5 in the previous chapter, where the ratio of average extra path length (i.e., α) is studied for grid networks and the results show that the average path of DSR is shorter than GSP. In fact, we transplanted the above simulation to a 10×10 static grid network, i.e., 100 transit nodes and 10 random deployed traffic nodes. The gossip sleep probability is 0.25 and the rest of parameters are same as in Section 3.5. After ten runs we got an average path length of 8.06 and 9.11 (hops) for DSR and GSP respectively.

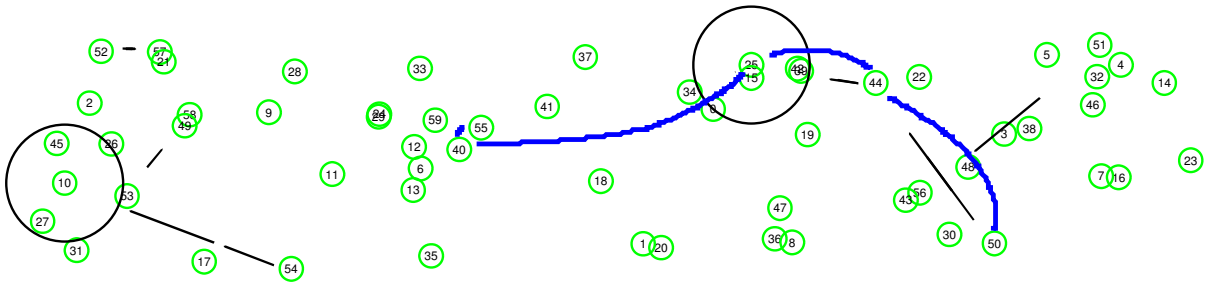

Figure 17: Snapshot of DSR at time 161

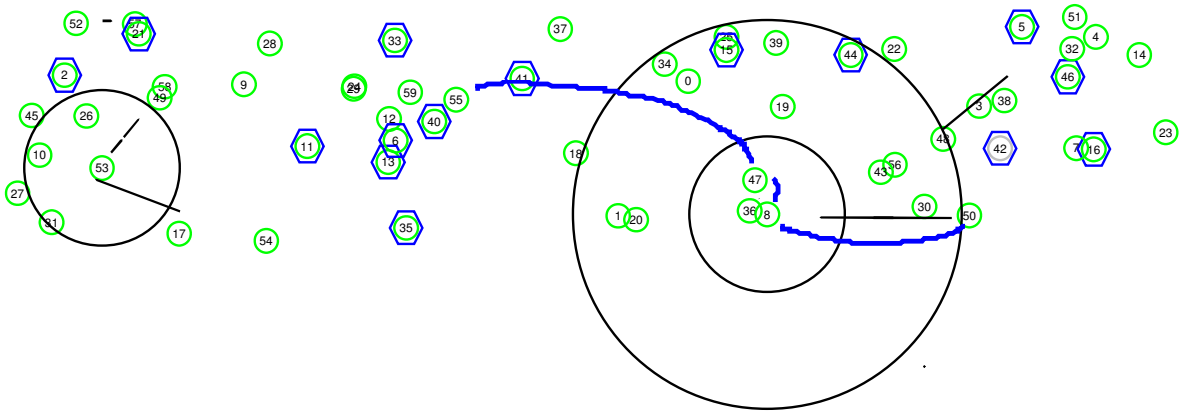

Figure 18: Snapshot of GSP at time 161

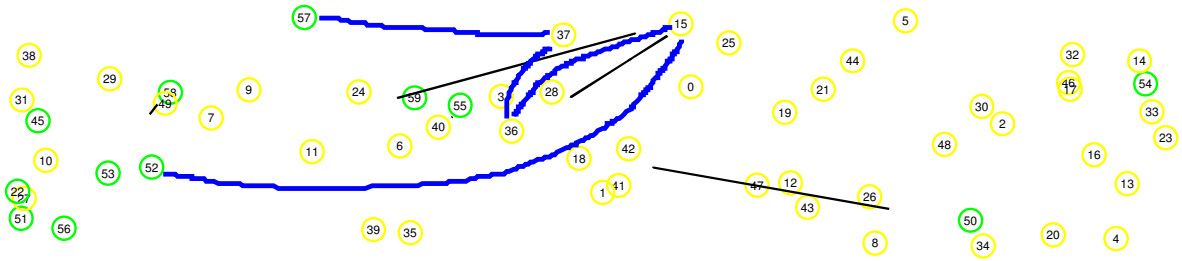

Figure 19: Snapshot of DSR at time 260

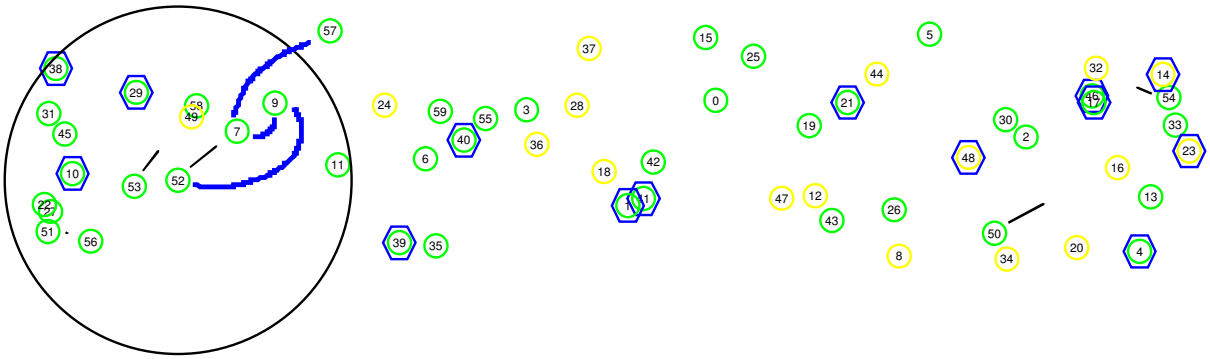

Figure 20: Snapshot of GSP at time 260

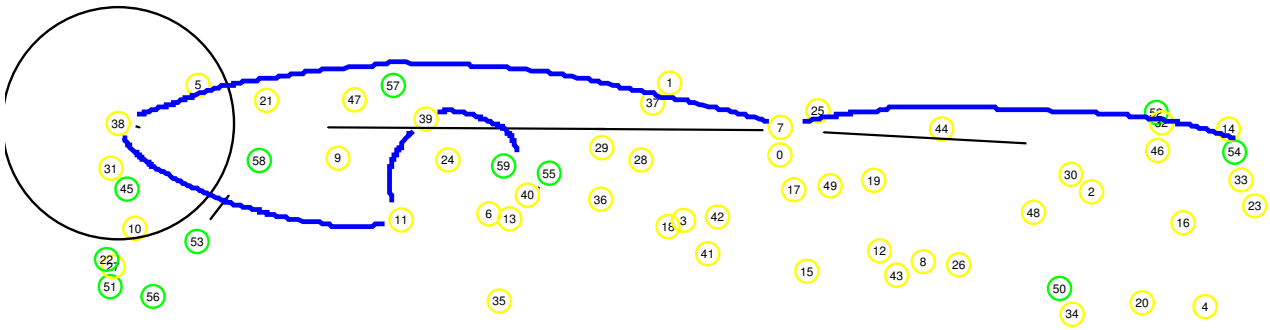

Figure 21: Snapshot of DSR at time 300

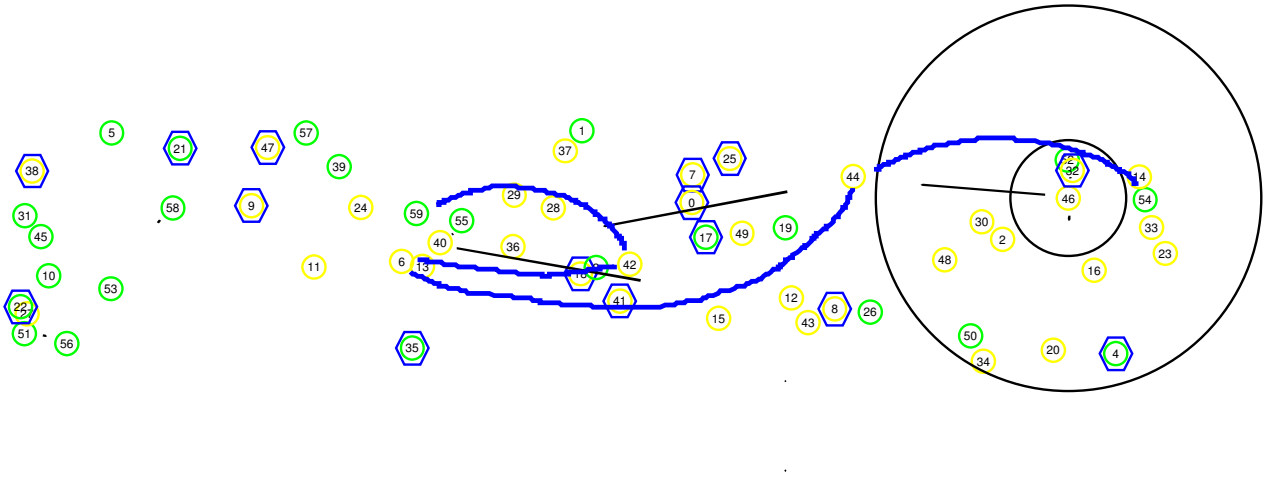

Figure 22: Snapshot of GSP at time 300

We summarize the parameter values used in Equation 4.3 in Table 8. Where, $B = 5(sources) \times 10(pkt/sec) \times 512(byte/pkt) \times 8(bit/byte) \div 2Mb/s = 0.1024$. We get the average path length L_{min} and L_{GSP} by deducting 1 from the simulation results to get rid of the traffic nodes' effects.

We put these values in Equation 4.3 to get the analytical lifetime improvement of GSP. For the 50-node and 100-node networks, we get, $\mathcal{L} = 0.427298$ and $\mathcal{L} = 0.998432$ respectively.

From Equation 4.2, we get $\mathcal{L}_{GSP}$ as 570.9192(sec) and 799.3728(sec) for 50 and 100 node networks respectively. We compare these two values with simulation results, as shown in Fig. 23, 24, and 25. The definition of lifetime in the analytical model is clear, i.e., the time when all the nodes die out and they die at almost the same time due to the assumption that there are a very large number of nodes and the gossip sleep period is much smaller than the network lifetime. This is not the case in our GSP simulation and nodes die at different times. However, we can roughly use the mid point of the decay part of the curves as the network lifetime in our simulation, i.e., the time when half of the nodes are alive and the other half are dead. We can see that the analytical results are just slightly larger than the simulation results due to the fact that we didn't include the routing overhead and MAC layer overhead in the analytical model.

Table 8: Parameter Values Used in the Analytical Model

Parameter	Value
P_{tx}	1.327(W)
P_{idle}	0.844(W)
B	0.1024
N	50/100
L_{min}	1.88195713/1.396316
L_{GSP}	1.85242994/1.369098
$\mathcal{L}_{non}$	400(sec)

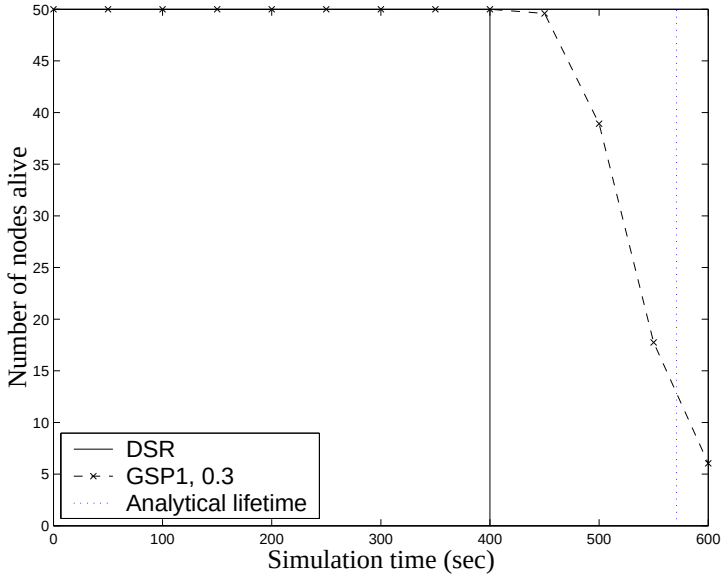

Figure 23: Network lifetime comparison between simulation and analytical results in a network of 50 transit nodes with gossip sleep probability p = 0.3

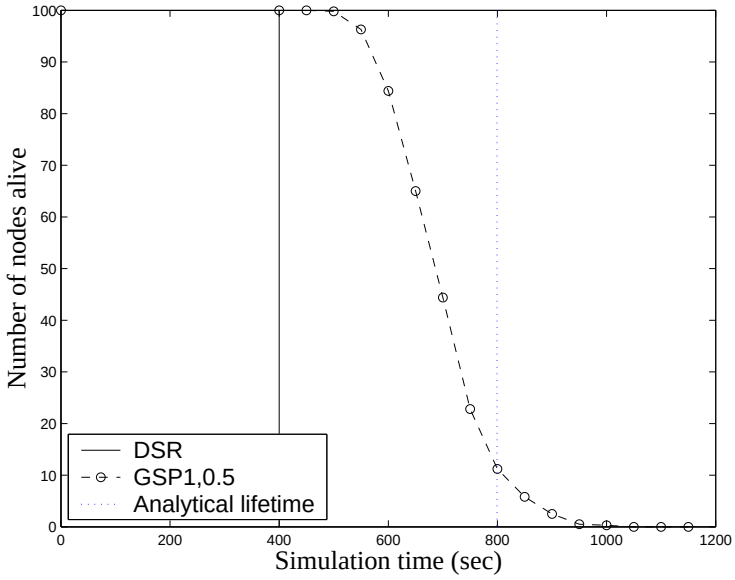

Figure 24: Network lifetime comparison between simulation and analytical results in a network of 100 transit nodes with gossip sleep probability p = 0.5

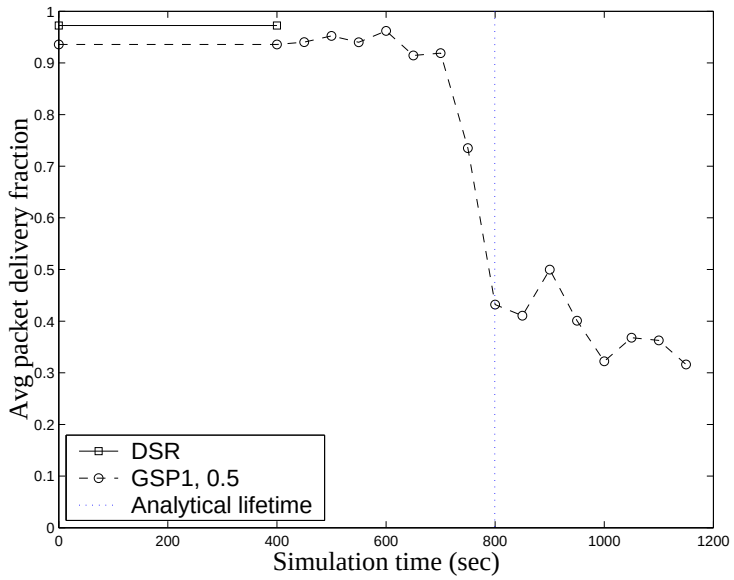

Figure 25: Network lifetime comparison between simulation (packet delivery fraction) and analytical results in a network of 100 transit nodes with gossip sleep probability p = 0.5

4.2 COMPREHENSIVE SIMULATION

In this section, we conduct a more complete simulation for more scenarios. Various simulation parameters are changed to get a better understanding of GSP performance.

4.2.1 Parameters

We continue to use DSR as the routing protocol to study the performance of GSP and we focus on the 100 node network (i.e., 100 transit nodes and 10 traffic nodes). The parameters we study here and their values are as follows:

- Gossip sleep probability(p): 0.5, 0.6, 0.7
- Traffic load: 5, 10, 15, 20 pkt/sec/flow
- Pause time: 0, 30, 60, 90, 120, 300, 600, static (i.e., infinity) seconds

In other words, we want to study the performance of GSP under different sleep probabilities, traffic loads, and mobility. These are the major factors which may affect the performance of GSP significantly in addition to the node density, of which the effects can be clearly observed in Section 4.1. Again, the values of these parameters are chosen to study the properties of GSP and they are not necessarily the optimal values.

4.2.2 Simulation Results

The simulation follows the model in Section 4.1.1 unless otherwise stated. Here, we just study the performance of Asynchronous GSP, i.e., GSP2, which is a more generic version of GSP. The performance metrics we evaluate here are packet delivery fraction and end-to-end delay. Every curve in the figures is an average of 10 to 20 simulation runs.

4.2.2.1 Packet Delivery Fraction Fig. 26– 29 show the average packet delivery fraction of DSR versus simulation time in various scenarios. We use them as a reference to study the performance of GSP. Fig. 30– 41 show the average packet delivery fraction of GSP2 in different scenarios. Many curves in the figures are almost overlapped, however, we still can see the performance tendency given a condition. We can see in most cases, especially when the traffic load is

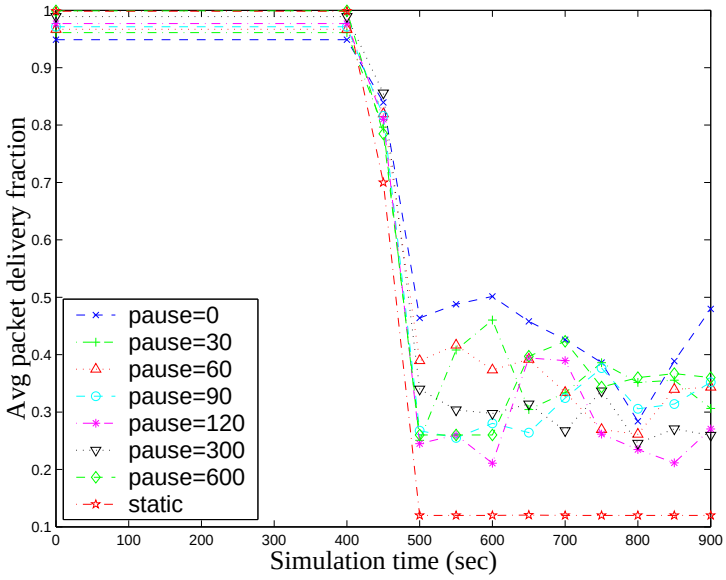

Figure 26: Average packet delivery fraction of DSR (traffic load=5pkt/sec/flow) in the network of 100 transit nodes with different mobility level (pause time)

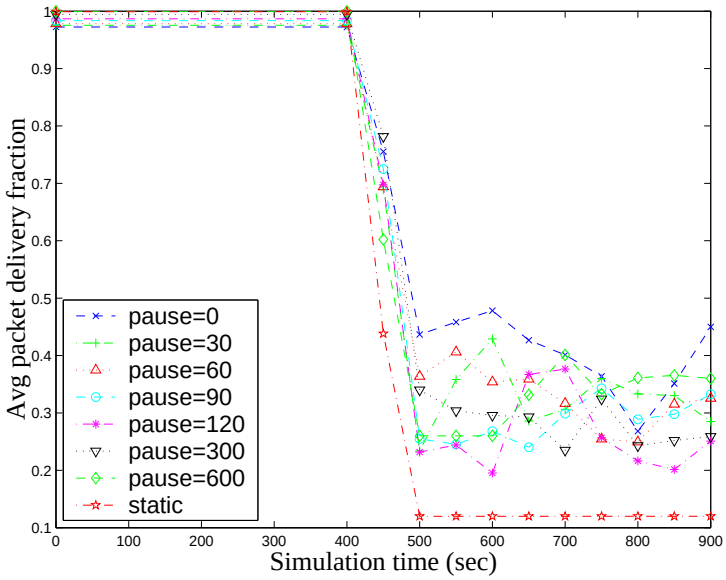

Figure 27: Average packet delivery fraction of DSR (traffic load=10pkt/sec/flow) in the network of 100 transit nodes with different mobility level (pause time)

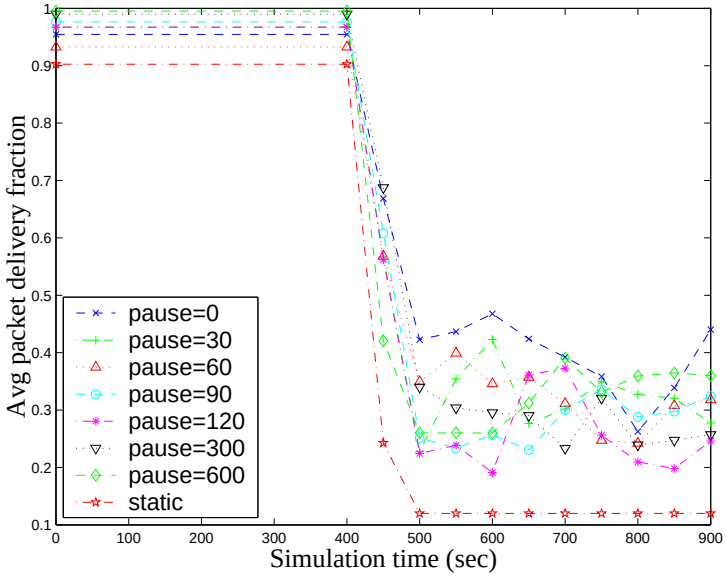

Figure 28: Average packet delivery fraction of DSR (traffic load=15pkt/sec/flow) in the network of 100 transit nodes with different mobility level (pause time)

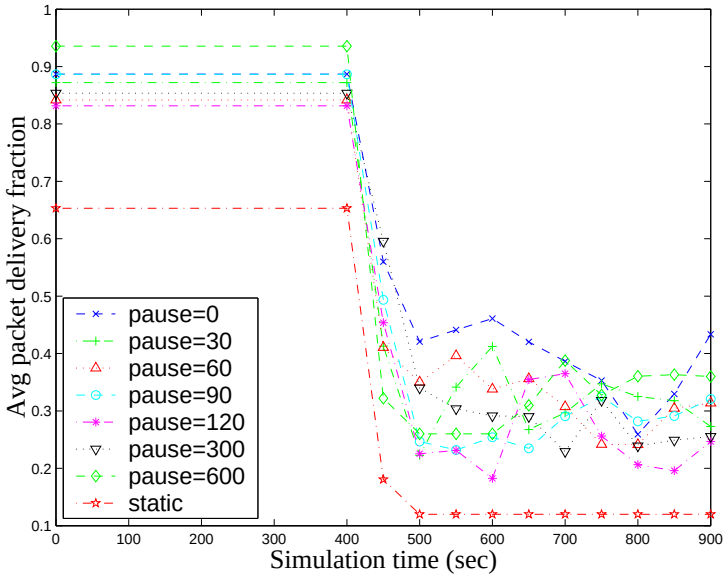

Figure 29: Average packet delivery fraction of DSR (traffic load=20pkt/sec/flow) in the network of 100 transit nodes with different mobility level (pause time)

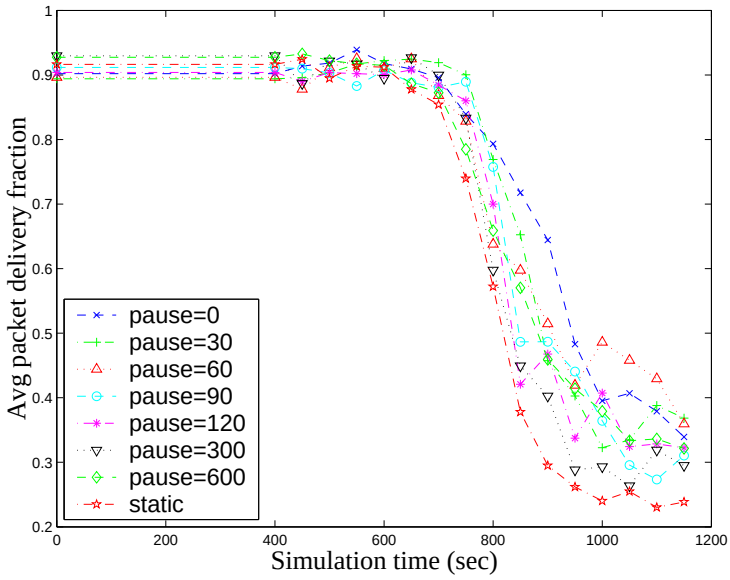

Figure 30: Average packet delivery fraction of GSP2 (gossip sleep probability $p = 0.5$, traffic load=5pkt/sec/flow) in the network of 100 transit nodes with different mobility level (pause time)

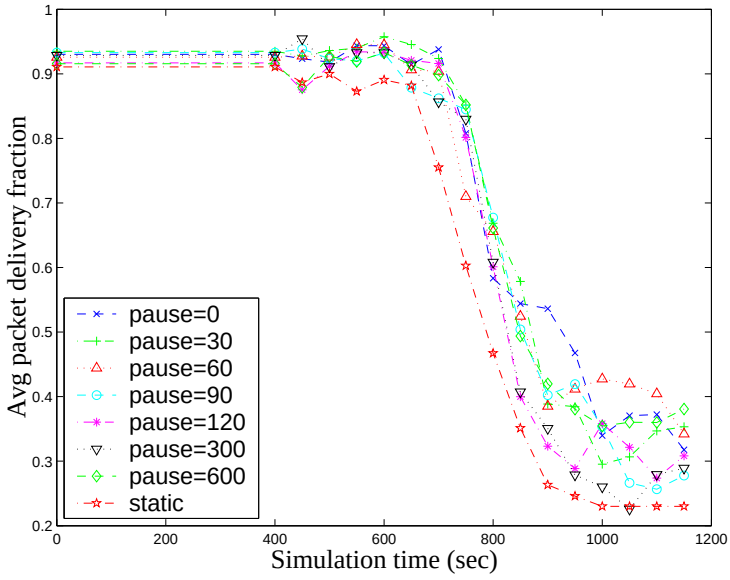

Figure 31: Average packet delivery fraction of GSP2 (gossip sleep probability $p = 0.5$, traffic load=10pkt/sec/flow) in the network of 100 transit nodes with different mobility level (pause time)

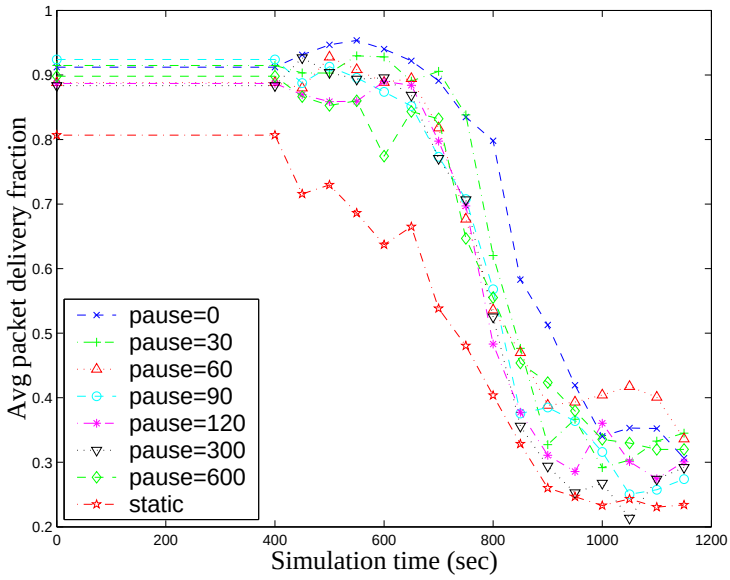

Figure 32: Average packet delivery fraction of GSP2 (gossip sleep probability $p = 0.5$, traffic load=15pkt/sec/flow) in the network of 100 transit nodes with different mobility level (pause time)

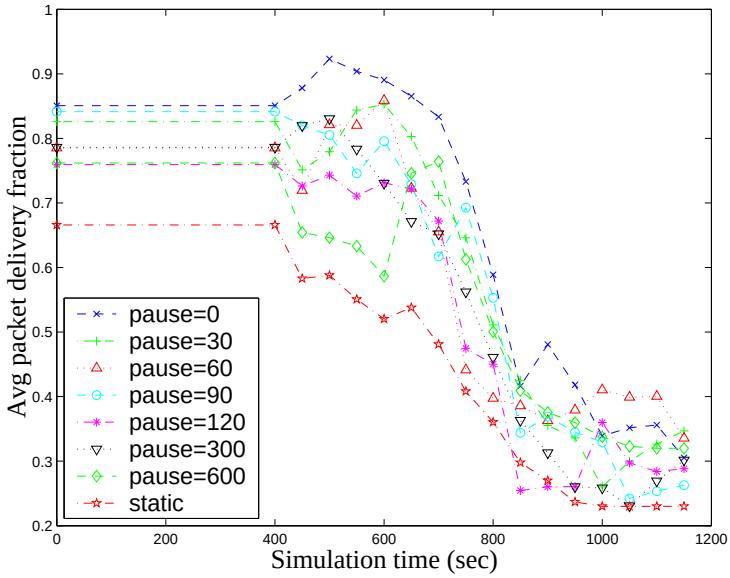

Figure 33: Average packet delivery fraction of GSP2 (gossip sleep probability $p = 0.5$, traffic load=20pkt/sec/flow) in the network of 100 transit nodes with different mobility level (pause time)

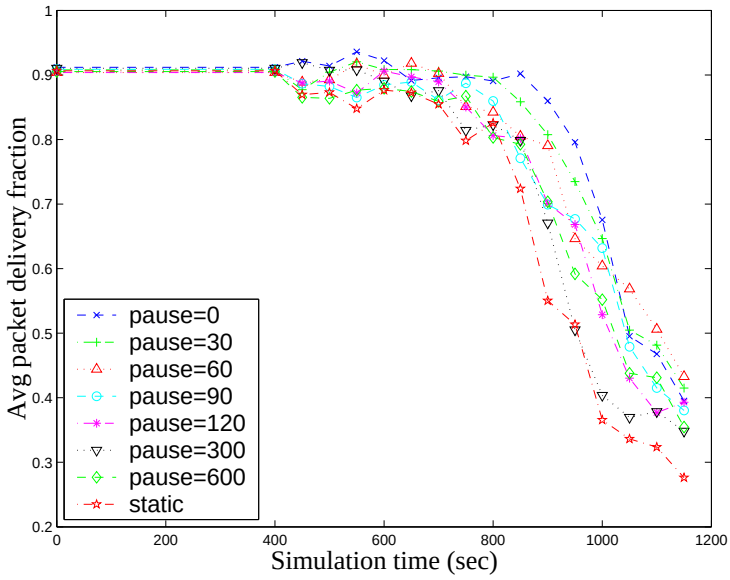

Figure 34: Average packet delivery fraction of GSP2 (gossip sleep probability $p = 0.6$, traffic load=5pkt/sec/flow) in the network of 100 transit nodes with different mobility level (pause time)

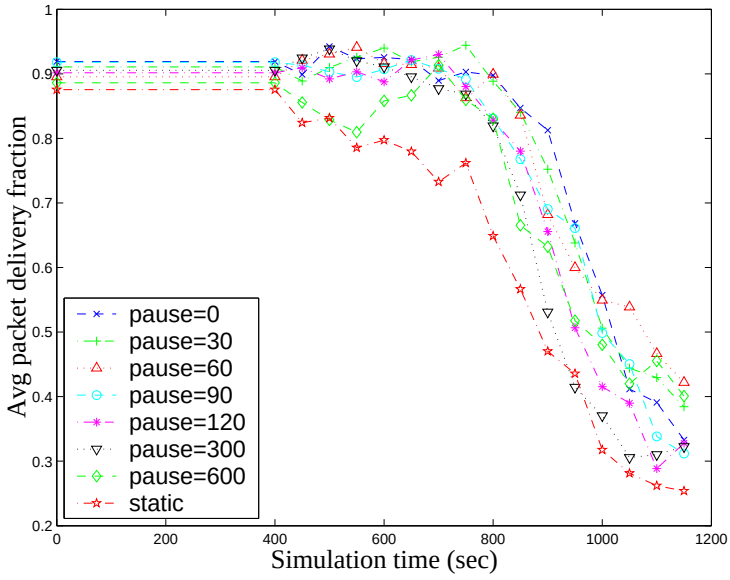

Figure 35: Average packet delivery fraction of GSP2 (gossip sleep probability $p = 0.6$, traffic load=10pkt/sec/flow) in the network of 100 transit nodes with different mobility level (pause time)

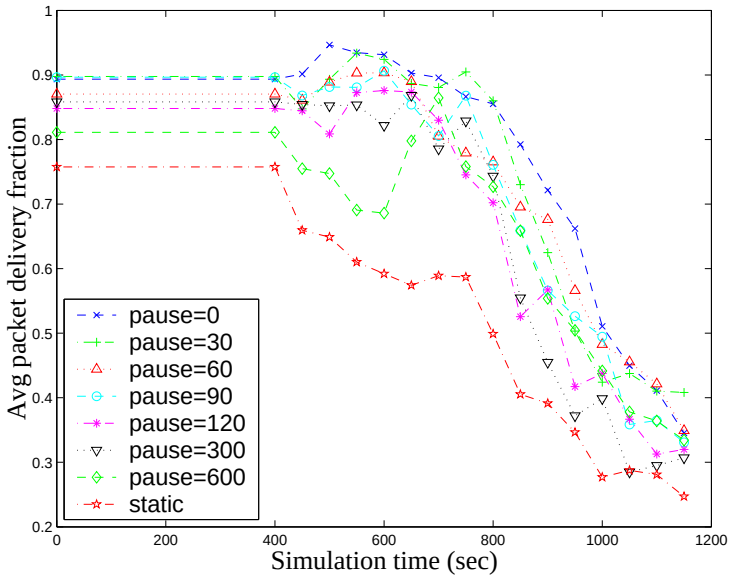

Figure 36: Average packet delivery fraction of GSP2 (gossip sleep probability $p = 0.6$, traffic load=15pkt/sec/flow) in the network of 100 transit nodes with different mobility level (pause time)

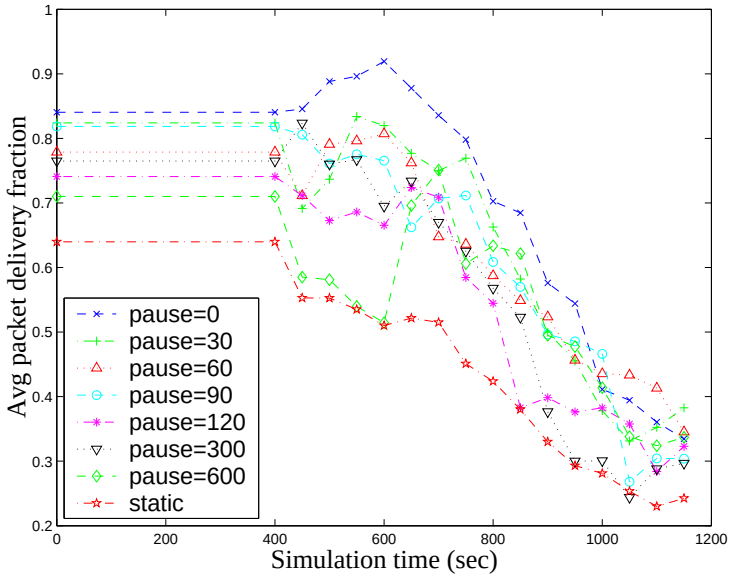

Figure 37: Average packet delivery fraction of GSP2 (gossip sleep probability $p = 0.6$, traffic load=20pkt/sec/flow) in the network of 100 transit nodes with different mobility level (pause time)

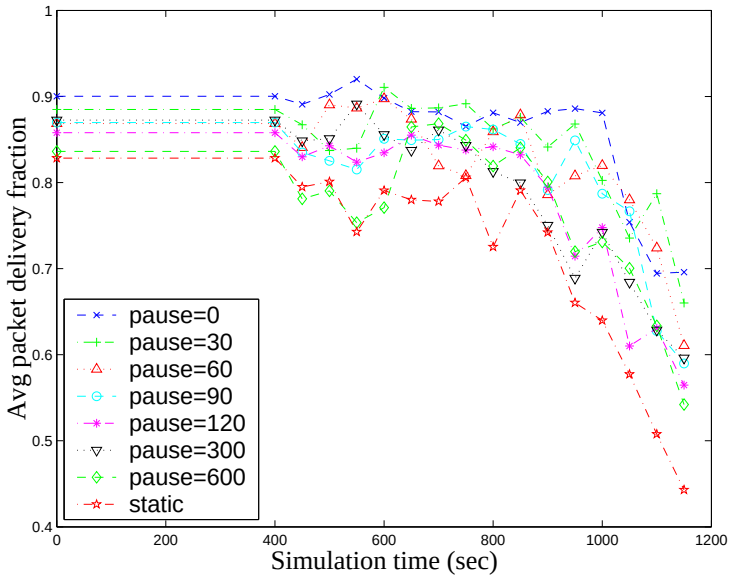

Figure 38: Average packet delivery fraction of GSP2 (gossip sleep probability $p = 0.7$, traffic load=5pkt/sec/flow) in the network of 100 transit nodes with different mobility level (pause time)

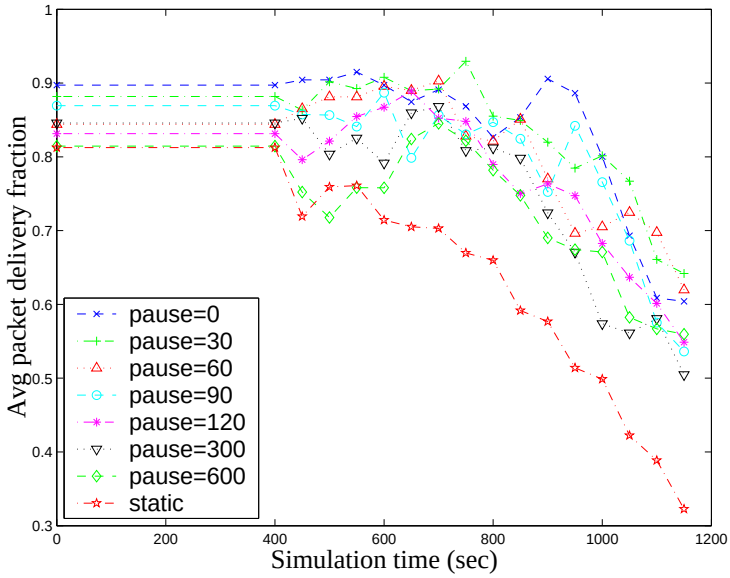

Figure 39: Average packet delivery fraction of GSP2 (gossip sleep probability $p = 0.7$, traffic load=10pkt/sec/flow) in the network of 100 transit nodes with different mobility level (pause time)

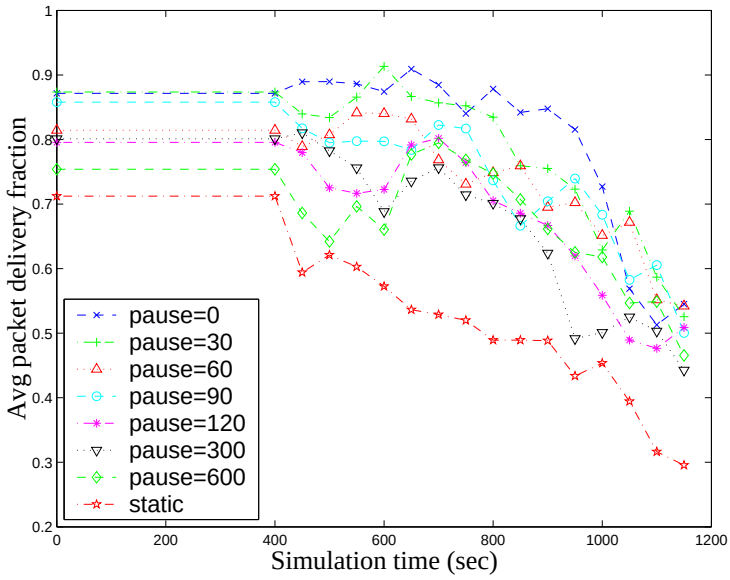

Figure 40: Average packet delivery fraction of GSP2 (gossip sleep probability $p = 0.7$, traffic load=15pkt/sec/flow) in the network of 100 transit nodes with different mobility level (pause time)

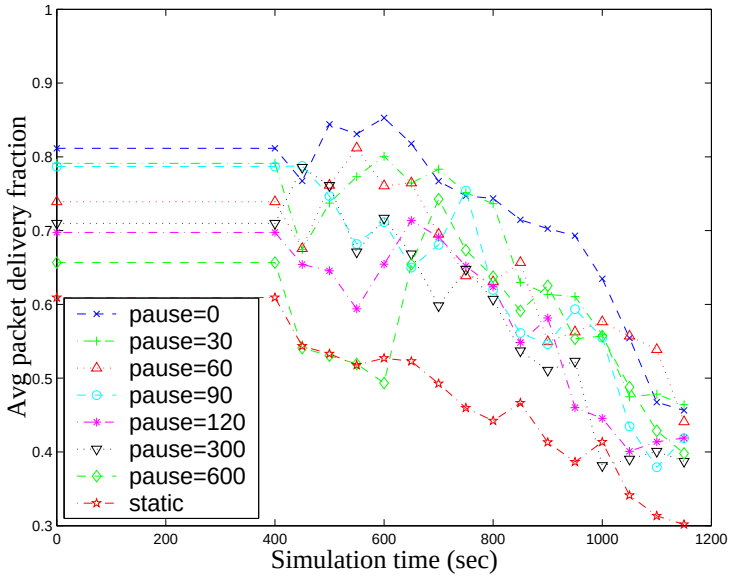

Figure 41: Average packet delivery fraction of GSP2 (gossip sleep probability $p = 0.7$, traffic load=20pkt/sec/flow) in the network of 100 transit nodes with different mobility level (pause time)

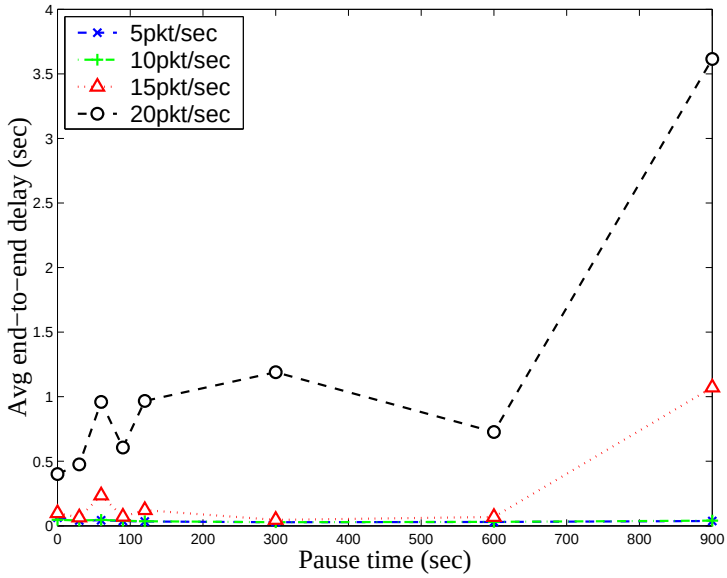

Figure 42: Average end-to-end delay of DSR with various traffic load versus nodal pause time in the network of 100 transit nodes

high, the lower mobility generates lower packet delivery fraction. The reason is that the mobility can increase the capacity of ad hoc networks [19].

When the network is static, the performance is much lower in case of high traffic. The reason is that the node speed would affect the average node degree of a network in the Random Waypoint model, i.e., higher node speed, higher average node degree [10]. In other cases, although the pause time is different, the average node speed is same. With a lower average node degree, the static network performance decreases when the traffic increases.

The rest of the results are along the lines of what we expected: higher sleep probability, worse average packet delivery performance but longer network lifetime; higher traffic load, worse performance in both DSR and GSP.

4.2.2.2 End-to-end Delay Fig. 42– 46 show the average end-to-end delay of the first 400 seconds of DSR and GSP versus the mobility level (i.e., the pause time). The most notable phenomenon is the curves of traffic load 20 pkt/sec, which are higher than other traffic loads most of the

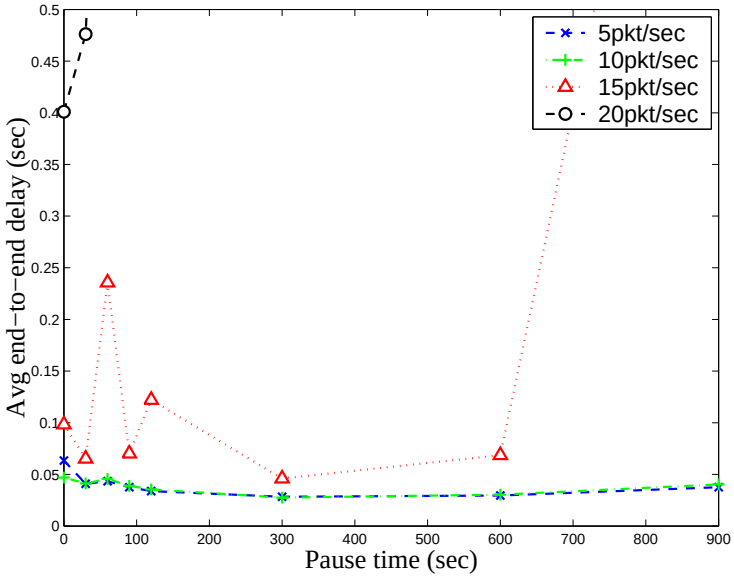

Figure 43: Zoom of Fig. 42

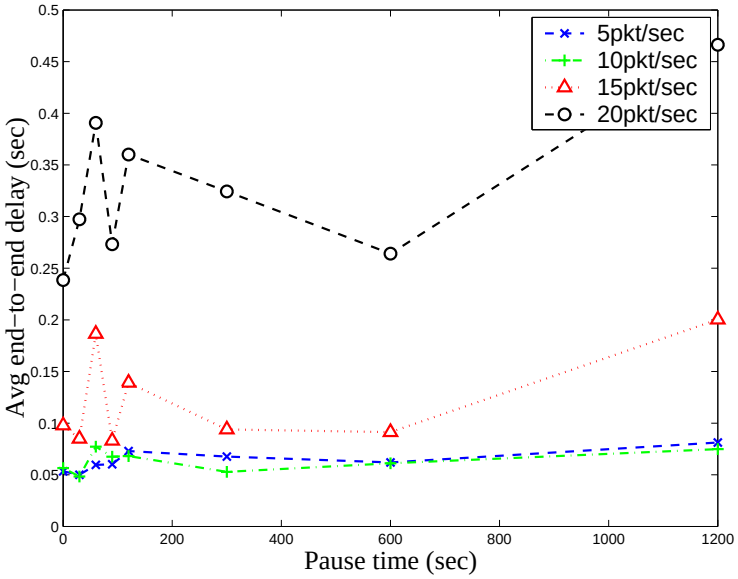

Figure 44: Average end-to-end delay of GSP2 (gossip sleep probability $p = 0.5$) with various traffic load versus nodal pause time in the network of 100 transit nodes

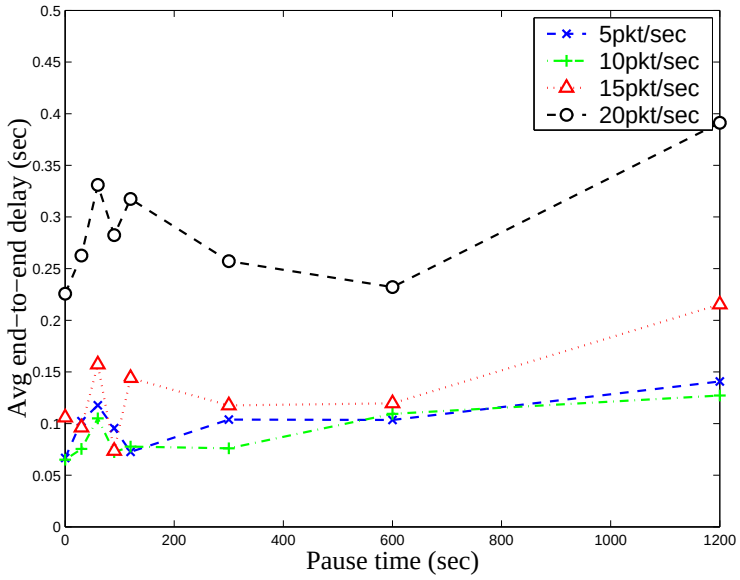

Figure 45: Average end-to-end delay of GSP2 (gossip sleep probability $p = 0.6$) with various traffic load versus nodal pause time in the network of 100 transit nodes

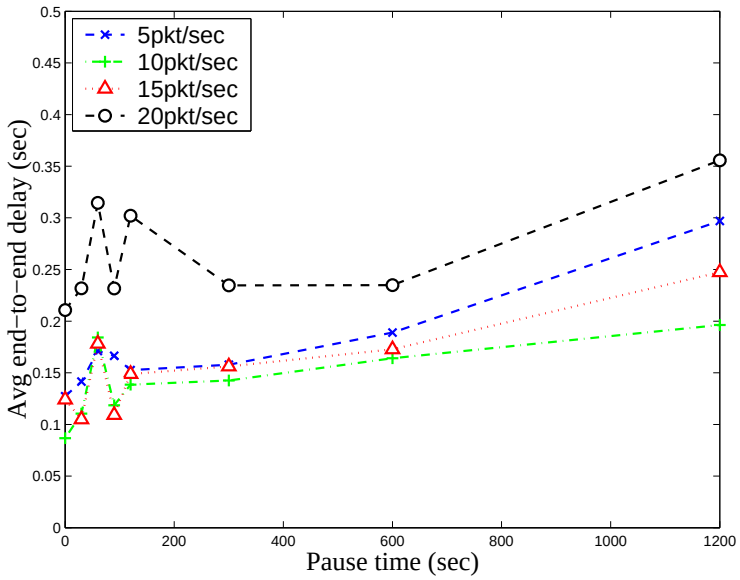

Figure 46: Average end-to-end delay of GSP2 (gossip sleep probability $p = 0.7$) with various traffic load versus nodal pause time in the network of 100 transit nodes

time in all the figures. The traffic is too high for the network and the packets have to wait in the intermediate nodes for a long time. The static network capacity is particularly low, so the delay is higher than other cases. Among these high traffic load scenarios, the curve of DSR in Fig. 42 is particularly high, which counterattacks our intuition that DSR always has a better performance due to the lack of path failures incurred by GSP. Actually, when the traffic is very high, the path failure may remove the packets using the path from a node's buffer, making other packets get through faster. This may not necessarily decrease the packet delivery fraction since the packets may be dropped anyway after the buffer is full.

4.3 DISCUSSION

Based on the simulation results obtained in the previous sections, we can draw some heuristics to adjust the value of p in this section. The purpose is to qualitatively understand the selection of parameter p given a network scenario. In addition, this may help us develop extended GSP protocols to improve and adapt the basic GSP in the next Chapter.

Heuristic 1: Increase p if a network has higher density, decrease otherwise.

Heuristic 1 is a direct observation of the results in Section 4.1, i.e., in a denser network like the 100 node network, we can use larger sleep probability like $0.5, 0.6, 0.7$. This heuristic can be generalized to local density, i.e., the nodes in a dense area of a network can increase p. We will further discuss this idea in Section 5.1

Heuristic 2: Decrease p when traffic increases, increase otherwise.

GSP places nodes in a sleep state, which may remove some of the paths existing between a source and a destination and decrease the capacity of a network. If we want the network to handle more traffic, we need to increase p to preserve more capacity. From the previous simulation results, we can see that the network performs better if we use a small p when the traffic is high. Of course,

the network lifetime may be reduced.

Heuristic 3: Decrease p when mobility decreases, increase otherwise.

As we know, the mobility can improve the network capacity. The extreme case is a static network, where the capacity is much lower than a mobile network due to the property of Random Waypoint model, and thus much more sensitive to traffic increase. If a real application follows the Random Waypoint model or any model with the similar property, we should design the network with extra caution in choosing the value of p.

Heuristic 4: Decrease p if a low end-to-end packet delay is required.

The gossiping technique may break an active path and interrupt an on-going traffic flow. This makes the routing protocol start a route repair or discovery process, which may take a long time. Therefore, to reduce the delay, we should limit the usage of gossiping, i.e., decreasing p. This heuristic can also be generalized to local usage, i.e., we can just decrease the p of the nodes carrying traffic, as we will discuss in Section 5.3.

4.4 IMPACTS OF ROUTING PROTOCOLS

In this section, we discuss the possible impacts of different ad hoc routing protocols. This can give us a general idea of how to integrate GSP with different routing protocols and what network features we may expect.

4.4.1 Proactive Protocols

As reviewed in Chapter 2, proactive routing protocols maintain consistent, up-to-date routing information from each node to every other node in the network. They store route information all the time, even before it is needed. Therefore, every topology change will trigger a network-wide

routing information update, which may incur heavy control traffic in the network. With GSP, in addition to the node mobility, the node's status change between an active mode (transmit, receive, idle) and a sleep mode would almost surely trigger such an update. In other words, the status change of a node may affect the entire network. This is the drawback of proactive protocols and GSP makes it worse.

On the other hand, the major advantage of proactive protocols, i.e., keeping track of routes for all destinations in the network all the time, can mitigate the impacts of GSP on the network performance. A status change of a node can be broadcasted and treated quickly. A new path can be ready in a short time. Therefore, proactive protocols with GSP are expected to have better performance than reactive protocols with GSP when the traffic load is low, and worse when the traffic is high.

Based on the above analysis, we should follow the following rules when we design a network running a proactive routing protocol:

- Choose a larger gossip interval if the traffic is high, or the mobility is high. A larger gossip interval may reduce the frequency of status change, thus reduce the frequency of routing information update. This is very important when the high traffic load or mobility consumes a great portion of network bandwidth.
- Avoid choosing a medium p (e.g., 0.4, 0.5, 0.6). A medium p generates the highest status change frequency, thus the highest routing update information. On the other hand, both small and large p will make most of the nodes stay in their previous mode, i.e., fewer status change, thus fewer routing update.

4.4.2 Reactive Protocols

As we know, reactive routing protocols create and maintain routes only when desired by the source node. When a node requires a route to a destination, it initiates a route discovery process, typically, by flooding. Therefore, the major features of reactive protocols are less routing updates and long initial delay. We can expect that the impact of GSP on reactive routing updates should be less than on proactive routing updates. The status change of nodes not involved in traffic transmission in the network does not incur any routing updates. The number of this type of nodes, of course, depends

on the traffic load. Furthermore, placing some nodes in sleep mode may reduce the neighbor information exchange required by some reactive protocols, for instance, AODV [44] [45]. The flooding overhead of route discovery may also be reduced.

On the other hand, GSP may incur more packet drops and a longer delay. A network may face more path failures with GSP when the nodes on active paths switch to sleep mode. After a path failure, two actions may be taken by reactive protocols. First, the routing protocol may try to repair the path by avoiding the failed nodes. This process may incur less delay and save the packets en route. Compared with proactive protocols, the chance to repair the broken path is low since the intermediate nodes may not maintain an alternate path to the destination. If the repair process fails, the routing protocol has to start the second action, i.e., a new route discovery process, to find an alternate path from the source to the destination. This process takes more time, generates more overhead, and typically drops all packets en route. Therefore, GSP may incur more packet loss and end-to-end delay.

The following are the general design rules of GSP for reactive routing protocols:

- Compared with proactive protocols, we can choose a relatively larger p if the traffic is low, especially, if the number of traffic flows is low. The reason is that, with this type of traffic, the number of the nodes involved in traffic transmission is low, hence, the routing overhead incurred by GSP is low.

- Employ a route repair process in routing protocol. Some reactive routing protocls have this option. For example, DSR provides the salvage process to repair a broken path. Users can decide whether or not use it depending on their application. As discussed above, this type of function can reduce the packet delay and loss.

- If possible, choose a smaller p or larger gossip interval for the nodes involved in traffic transmission. This can reduce the path failures incurred by GSP while maintain a higher average p or smaller average gossip interval in the network. This approach is discussed in Chapter 5.

4.5 SUMMARY

In this chapter, we conduct a set of simulations to study the performance of GSP. The simulation results agree with the analytical results in the previous chapter. From the simulation results, we can see that certain values of p make almost all the awake nodes in the network connected. On the other hand, the performance of the network is only slightly affected. Another advantage of GSP is that the energy consumption is more evenly distributed throughput the network since the nodes go to sleep in a fully random fashion and continuous traffic forwarding via the same path can be avoided.

Furthermore, We investigate the critical parameter p with subject to different network scenarios by a comprehensive simulation study. The network performance in terms of packet delivery fraction and end-to-end delay has been examined to explore the selection of p in certain network scenarios. In most cases, a smaller p brings better performance. The selection of p is also affected by the routing protocols with which the GSP is integrated. In general, we should consider the trade-off of routing overhead and initial delay between proactive and reactive routing protocols. Sensor network protocols are basically reactive. How to select the optimal value of p theoretically is a topic of future work.

5.0 EXTENSIONS OF GSP

In this Chapter, we extend the basic GSP to heterogeneous networks with non-uniformly distributed node deployment and battery charge, in Section 5.1 and 5.2, respectively. In addition, we improve the performance of GSP by considering the ongoing traffic in Section 5.3. The simulation model used in this Chapter is same as the one in Section 4.1 unless otherwise stated. The summary and discussion are given in Section 5.4.

5.1 ADAPTIVE GSP (A-GSP)

GSP works well in a uniform deployed network, where the node density is the same everywhere. However, a fixed network wide value of p can not adapt to a network with non-uniform node density. A small sleep probability may keep a sparse part of the network connected but it will keep unnecessary nodes awake in a dense part of the network. Similarly, a large sleep probability may conserve more energy in the dense area, but partition the sparse area. To solve this problem, we propose the Adaptive Gossip-based Sleep Protocol (A-GSP) to dynamically adjust the gossip sleep probability (p) based on the local node density.

The local node density can be estimated based on the number of neighbors a node has. More neighbors means higher density, thus a small value of p should be employed. However, a node's p value should not be solely determined by the number of its neighbors. Otherwise, in a star topology, the central node may go to sleep with a large probability and cause a network partition. Here we propose that a node determines its sleep probability in conjunction with its neighbors. Specifically, node A suggests that its neighbors sleep with probability p_{nb}^{A} calculated from the following equation.

$$p_{nb}^{A} = \begin{cases} 0 & \text{if } N_{nb}^{A} \leq C, \\ 1 - C/N_{nb}^{A} & \text{if } N_{nb}^{A} > C. \end{cases}$$

where, N_{nb}^{A} is the number of node A's neighbors and C is a threshold. After collecting p_{nb}^{A} from all its neighbors, a node chooses the minimum as its own gossip sleep probability to satisfy all the neighbors.

A sketch of P_{nb}^{A} is shown in Fig. 47. We can see, with more neighbors, node A suggests its neighbors sleep more. Fig. 48 gives some examples with various values of C. A larger C makes the sleep probability increase slowly.

To make it clear, let's see two examples. Firstly, let's consider an infinite grid topology, with each node having four neighbors. Therefore, $N_{nb}^{A} = 4$ is a constant and every node, say node A, suggests its neighbors sleep with probability $P_{nb}^{A} = 1 - C/4$ if $C < 4$. For example, $P_{nb}^{A} = 0.5$ if $C = 2$ and $P_{nb}^{A} = 0.25$ if $C = 3$. Secondly, let's consider a random topology, shown in Fig. 49. Additionally, we assume $C = 2$. Node A has 6 neighbors, i.e., node B, C, D, E, F, and G. It recommends they sleep with probability $P_{nb}^{A} = 1 - 2/6 = 2/3$. This is the only value and therefore the minimal value node B, C, D, and E can get, so they set their gossip sleep probability $p = 2/3$. On the contrary, node A is the only neighbor of node B, C, D, and E. We can see the minimal value node A can collect from its neighbors must be 0 and node A has to stay awake all the time. The rest of the topology, i.e., node F, G, H, I, J, K, and L should set their gossip sleep probability to $p = 1/3$. The reason is they all have at least one neighbor with its degree equal to 3, which happens to be the smallest degree node and makes the sleep probability this node suggests the smallest one.

A-GSP needs neighbor information to adaptively adjust the gossip sleep probability. Node A recalculates p_{nb}^{A} right before it broadcasts itself and attaches the updated p_{nb}^{A} to the message. Every node adjusts its own gossip sleep probability for every gossip interval. The neighbor information can be achieved with information already used in most routing protocols. Some of the most popular routing protocols are DSDV [43] and AODV [44]. Here, we use AODV as an example to describe how A-GSP works.

In the simulation, two types of topologies are used. One is a uniformly distributed network, where 100 transit nodes are randomly placed within a $1500m \times 300m$ area. The other is a non-uniformly distributed network, where 50 nodes are randomly placed within a $1500m \times 300m$ area

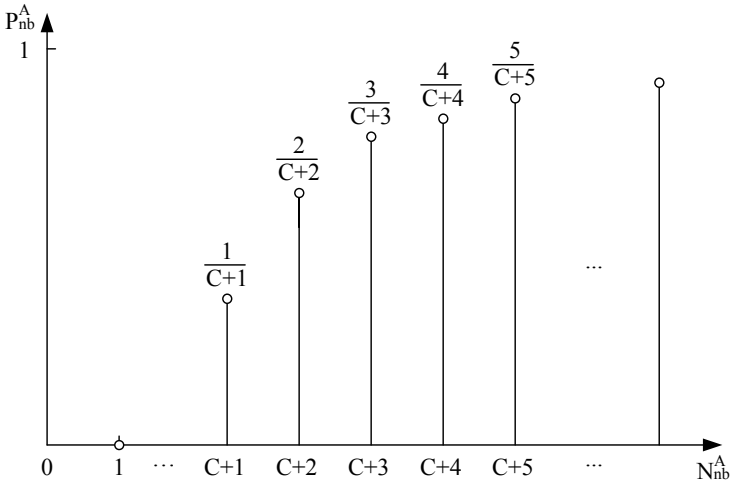

Figure 47: A sketch of P_{nb}^{A}

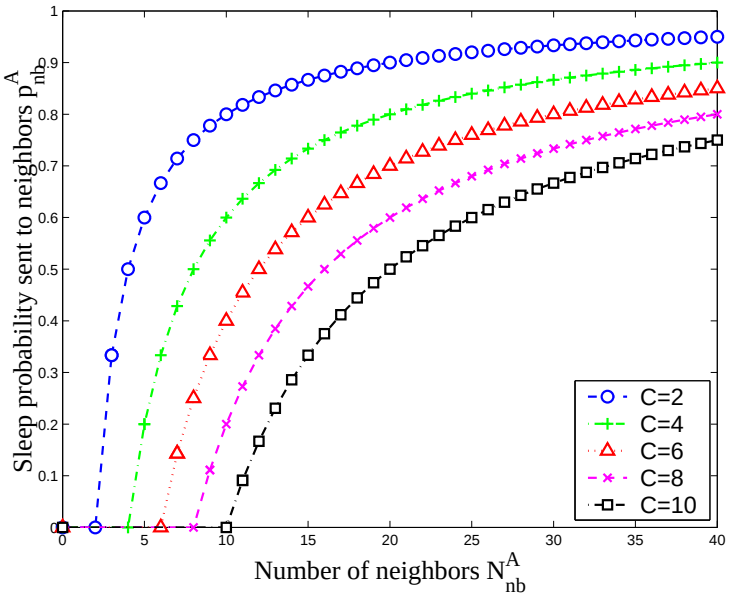

Figure 48: Examples of P_{nb}^{A}

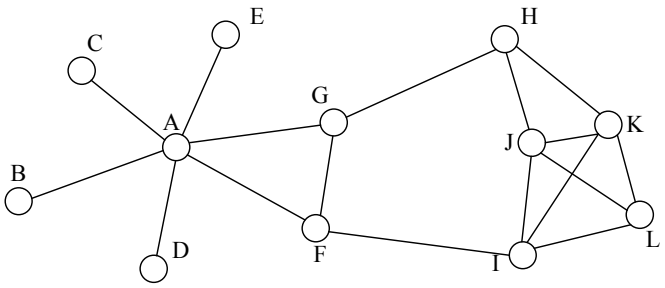

Figure 49: An example of random topology

and the other 50 in the left half of the area, i.e., $750m \times 300m$. We study the highest mobility case, i.e., pause time is 0. The transit nodes have enough energy so that the AODV protocol can run for 300 seconds.

The parameters of A-GSP are chosen to show the properties of A-GSP and they are not necessarily the optimal values. The gossip interval for both GSP and A-GSP is uniformly distributed between 0 and 40 seconds with an average of 20 seconds. The gossip sleep probability of GSP is varied as shown in the simulation result figures. We add 90% confidence intervals to the curves we want to differentiate.

Fig. 50 and Fig. 51 show the network lifetime and average packet delivery fraction of the uniformly distributed network. We can see that both GSP and A-GSP can successfully extend the network lifetime and maintain an acceptable network performance. In the simulation, we use different values of gossip sleep probability (p) and the value of C in A-GSP is 3. In this case, they have a similar performance during their lifetime.

Fig. 52 and Fig. 53 show the network lifetime and average packet delivery fraction of the non-uniformly distributed network. In this scenario, although the network lifetime is similar to the uniformly distributed node case, the packet delivery fraction is much different. Since a portion of the network is sparse, a high sleep probability will hurt the network performance. For instance, when $p = 0.7$, although the number of alive nodes is large, the packet delivery fraction is much lower due to the fact that some part of the network is partitioned. In such a scenario, we prefer a better performance with a shorter network lifetime. The A-GSP with $C = 3$ achieved this

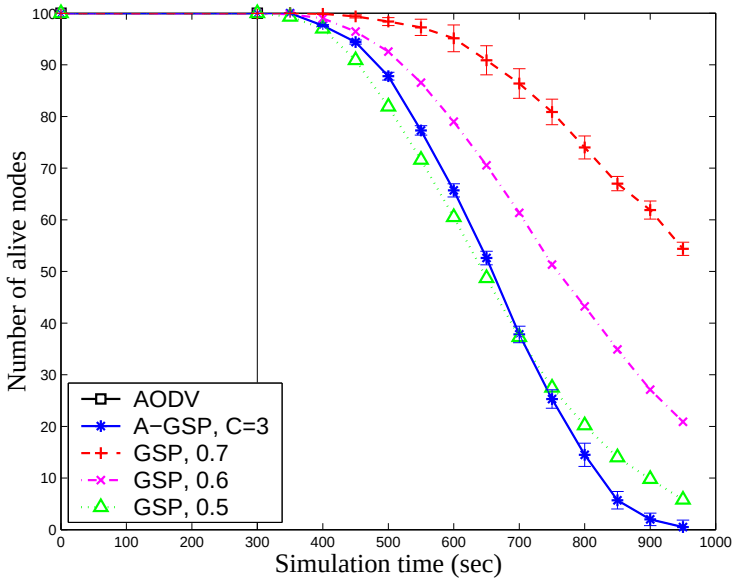

Figure 50: Network lifetime of AODV, A-GSP and GSP with different gossip sleep probability (p) in a uniformly distributed mobile network of 100 transit nodes, pause time=0

due to its adaptability. Therefore, A-GSP can adapt to both scenarios without manual parameter configuration, while GSP has to change its p value.

5.2 BATTERY-AWARE GSP (B-GSP)

The nodes in a network may have different battery lives. Making every node consume energy at the same rate may cause the nodes with low remaining energy to die quickly and lead to a partitioned network. Adapting to such networks requires us to take advantage of the information of a node's battery power. The design goal of Battery-aware GSP (B-GSP) is to prolong the entire network lifetime rather than the node lifetime.

The basic GSP makes every node consume its battery at the same rate no matter how much energy remains. A node with a larger amount of remaining energy will live longer. However, prolonging the entire network lifetime requires such a node to take more responsibility to help

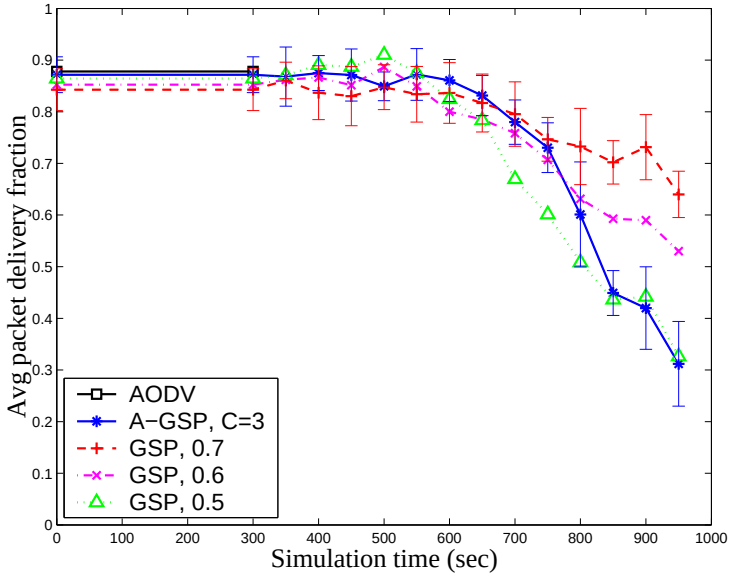

Figure 51: Average packet delivery fraction of AODV, A-GSP and GSP with different gossip sleep probability (p) in a uniformly distributed mobile network of 100 transit nodes, pause time=0

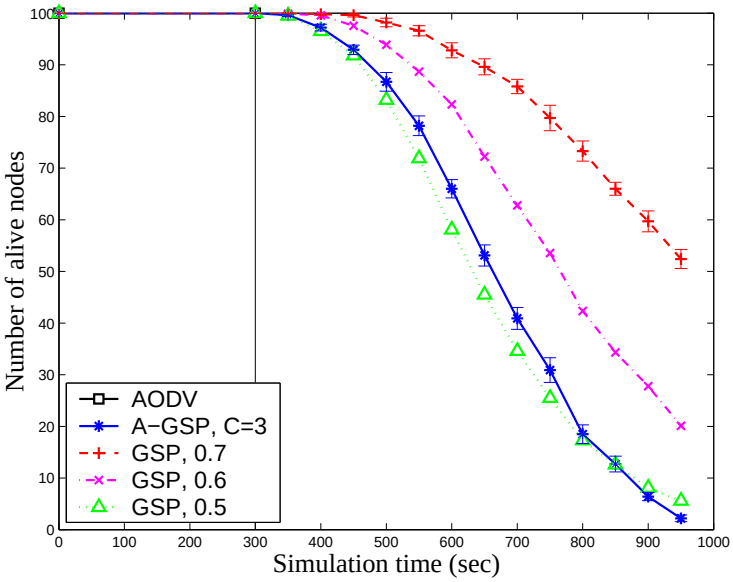

Figure 52: Network lifetime of AODV, A-GSP and GSP with different gossip sleep probability (p) in a non-uniformly distributed mobile network of 100 transit nodes, pause time=0

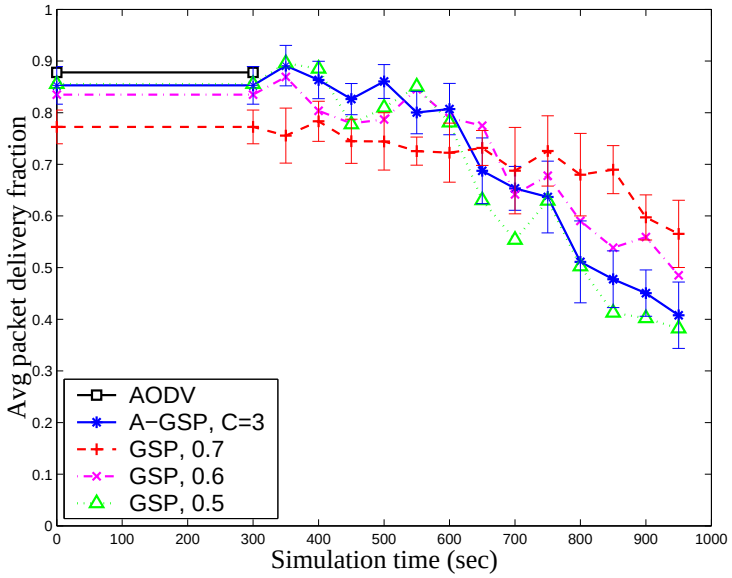

Figure 53: Average packet delivery fraction of AODV, A-GSP and GSP with different gossip sleep probability (p) in a non-uniformly distributed mobile network of 100 transit nodes, pause time=0

other nodes live longer. The question is how can a node know the overall battery dissipation rate, and adjust the gossip sleep probability, without global information. Here, we use a scheme to estimate this rate based on local knowledge, i.e., a node's one-hop neighbors' remaining battery capacity.

We assume every node knows its neighbors' remaining energy and dissipation rate, which can be achieved by $hello$ messages or other status exchange mechanisms. Let node i have remaining energy E_i, dissipation rate without GSP r_i, and gossip sleep probability p_i. The remaining lifetime of node i is $l_i = E_i/r_i$. The average remaining energy of node i's neighbors and itself is $\bar{E}_i$, and its average remaining lifetime is $\bar{l}_i$. The average dissipation rate of its neighbors and itself is $\bar{r}_i = \bar{l}_i/\bar{E}_i$. Also, we designate $\bar{p}$ the average gossip sleep probability of the entire network. Similar to p in the basic GSP, $\bar{p}$ is a given design parameter. To maximize the network lifetime with GSP, we want the lifetime of node i is equal to the average lifetime of its neighbors, i.e.,

$$\frac{\bar{l}_i}{\bar{q}} = \frac{l_i}{q_i} \tag{5.1}$$

where, $\bar{q} = 1 - \bar{p}$ and $q_i = 1 - p_i$. Therefore,

$$\frac{\bar{E}_i}{\bar{r}_i \times \bar{q}} = \frac{E_i}{r_i \times q_i} \tag{5.2}$$

$$q_i = \frac{E_i \times \bar{q} \times \bar{r}_i}{\bar{E}_i \times r_i} \tag{5.3}$$

Of course, if Equation 5.3 is greater than 1, we set $q_i = 1$. Like A-GSP, B-GSP also needs neighbor information to adjust the gossip sleep probability. Every node exchanges the remaining energy and dissipation rate with its neighbors via hello messages. Right after receiving a hello message, a node recalculates its own gossip sleep probability according to Equation 5.3. Although this new probability takes effect immediately, it will not affect the node's mode switching until the next gossip interval begins, i.e., the next time the node flips the coin. In addition, a node needs to recalculate its gossip sleep probability when the routing protocol concludes that it can not reach a neighbor any more, usually with no hello messages coming from that neighbor within certain time interval.

Again, we use AODV as an example in our simulation study to describe how B-GSP works. In the simulation, the initial energy can make half the nodes run for 100 seconds and the other half run for 300 seconds. We set the pause time of every node to 0, i.e., the highest mobility case is studied.

The parameters of B-GSP are chosen to show the properties of B-GSP and they are not necessarily the optimal values. The gossip interval for both GSP and B-GSP is uniformly distributed between 0 and 80 seconds with an average of 40 seconds. We assume every node has same dissipation rate. From Equation 5.3, we have

$$q_i = \frac{E_i \times \bar{q}}{\bar{E}_i} \tag{5.4}$$

In the simulation, we compare B-GSP with GSP when the gossip sleep probability is 0.7, i.e., $\bar{p} = 0.7 (\bar{q} = 0.3)$ in B-GSP and $p = 0.7$ in GSP.

Fig. 54 and Fig. 55 show the average packet delivery fraction and network lifetime of AODV, GSP and B-GSP with a 90% confidence interval. From Fig. 55, we can see AODV consumes half the nodes in the first 100 seconds and the other half in 300 seconds. Also, we can see that, although the basic GSP can prolong the network lifetime, it actually prolongs each node's lifetime with the

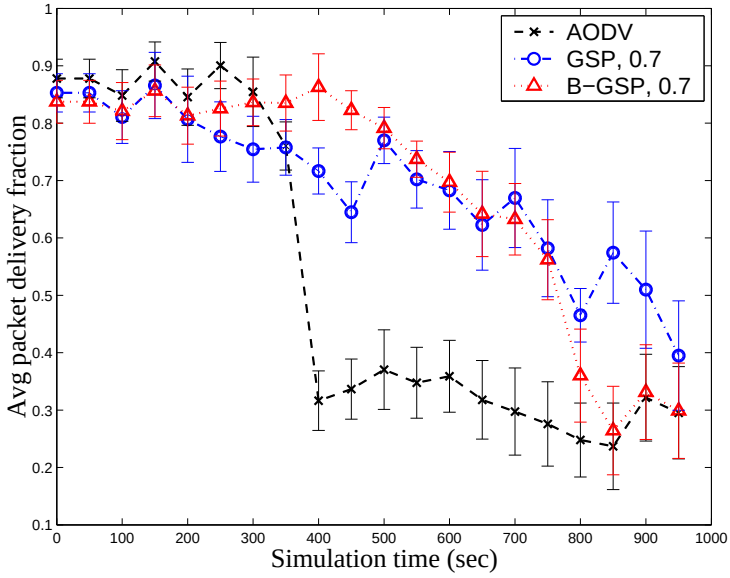

Figure 54: Average packet delivery fraction of AODV, GSP with $p = 0.7$ and B-GSP with $\bar{p} = 0.7$ in a network of 100 transit nodes with different initial energy, pause time is 0

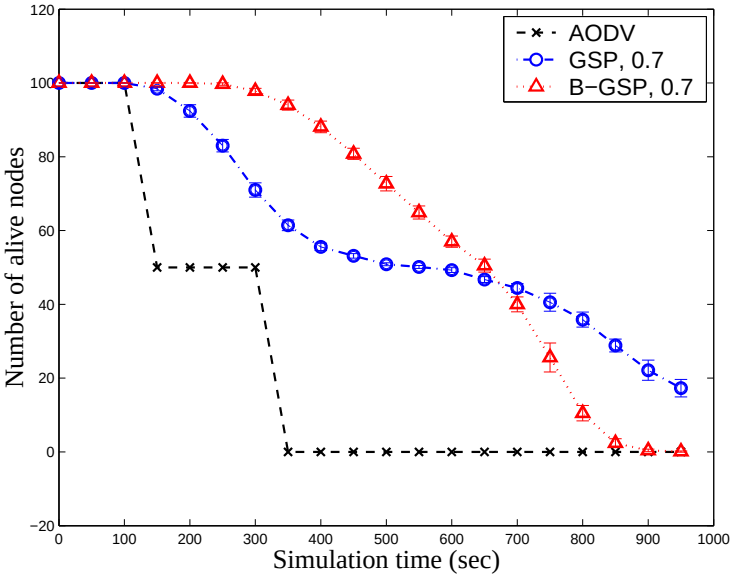

Figure 55: Network lifetime of AODV, GSP with $p = 0.7$ and B-GSP with $\bar{p} = 0.7$ in a network of 100 transit nodes with different initial energy, pause time is 0

same proportion. Only B-GSP can take advantage of all nodes' energy to maximize the entire network lifetime, i.e., try to make all nodes die around the same time. Fig. 54 shows the effects of different lifetimes on the network performance. It's quite obvious that only B-GSP can maintain the performance for the longest time.

5.3 TRAFFIC-AWARE GSP (T-GSP)

In this section, we improve the performance of GSP by considering the network traffic in the sleep decision. Putting nodes into a sleep mode may interrupt the ongoing traffic. This can result in a long packet delay, especially in a reactive routing protocol, where the setup delay of a new path is much longer than proactive protocols. Therefore, we propose the Traffic-aware Gossip-based Sleep Protocol (T-GSP) to reduce the probability of breaking an active communication when employing the gossiping technique.

The goal of T-GSP is to base energy management decisions on traffic patterns in the network. By reacting to changes in these patterns, nodes that carry traffic should stay awake with a higher probability while the other nodes can go to the sleep mode with the regular GSP sleep probability. The key idea of T-GSP is that transitions from active mode (i.e., send, receive or idle mode) to sleep mode should be triggered by a lack of active communication in addition to the gossiping probability. The lack of active communication is determined by a *traffic timer*, which is refreshed by the transmission/reception of packets at a node. The expiration of the timer of a node indicates that there may be no traffic using this node.

The value of the traffic timer can be determined by the type of packet received by a node. Different types of packets indicate different time intervals before the incoming of data transmission. Therefore, integration with a routing protocol and understanding the semantics of the received packets can help to determine the timer value. The basic concepts in determining a timer value discussed in [64] can be employed in our scheme. Flooded control packets provide a poor indication of subsequent data transmission. On the contrary, data packets usually indicate the possible arrival of the same type of packets to follow. Additionally, some special control packets, such as route reply packets in reactive routing protocols and query packets in sensor networks, provide a

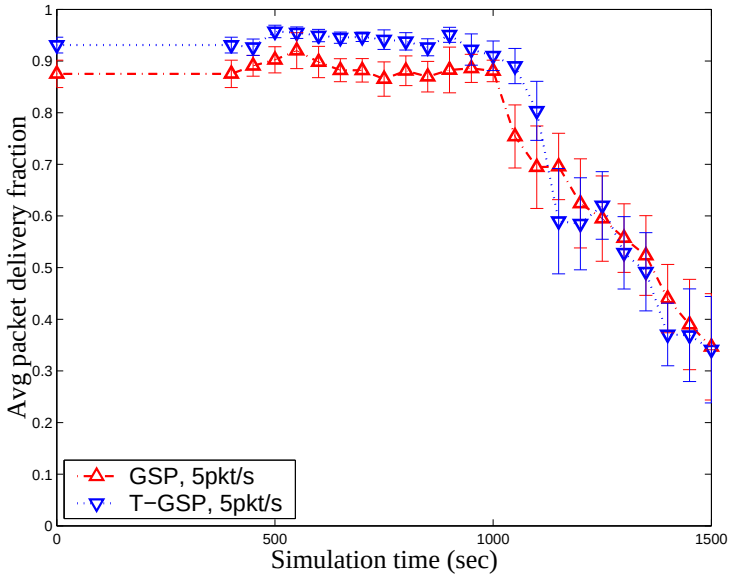

Figure 56: Average packet delivery fraction of GSP and T-GSP with long-lived CBR traffic (pause time=0, traffic rate=5pkt/sec)

good hint that subsequent data packets will follow this route. Therefore, the timer should be set to a value on the order of the data packet interarrival time, or the end-to-end delay between a source and a destination, whichever is greater. Since the packet interarrival time varies with traffic patterns, the timer value also varies with traffic patterns. Integrated with GSP, this scheme expects a gossip interval larger than the traffic timer value. This can be easily achieved since a gossip interval smaller than data packet interarrival time and end-to-end delay is not a good choice for GSP.

Before its traffic timer expires, a node should stay awake with a higher probability to reduce the possibility of breaking an active communication which is using this node. The nodes carrying no traffic can switch to sleep mode with a regular GSP sleep probability. We can see that a node requires no information from other nodes in T-GSP.

T-GSP can be integrated with a number of routing protocols. Here, we use DSR as an example to describe how T-GSP works. The traffic timer value is set as 1 second and a node will work for another 1 second with probability 1 after the gossip interval expires when the traffic timer has not. Other parameters of T-GSP are chosen to show the properties of GSP and they are not necessarily

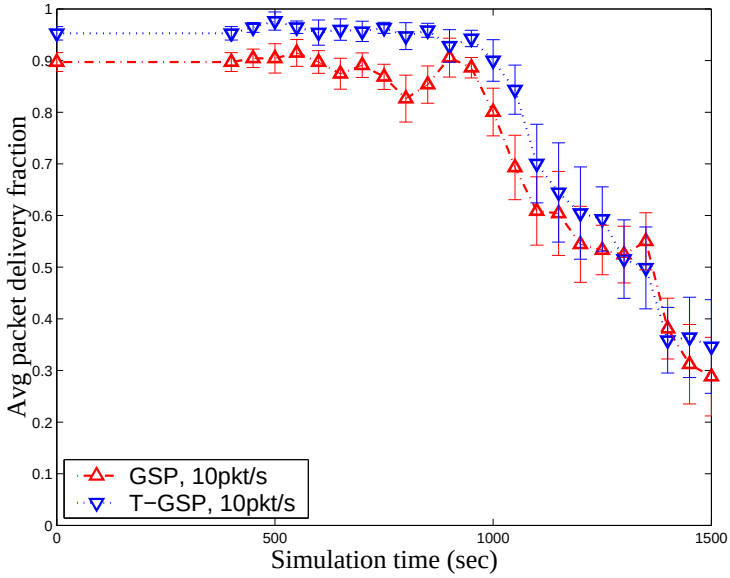

Figure 57: Average packet delivery fraction of GSP and T-GSP with long-lived CBR traffic (pause time=0, traffic rate=10pkt/sec)

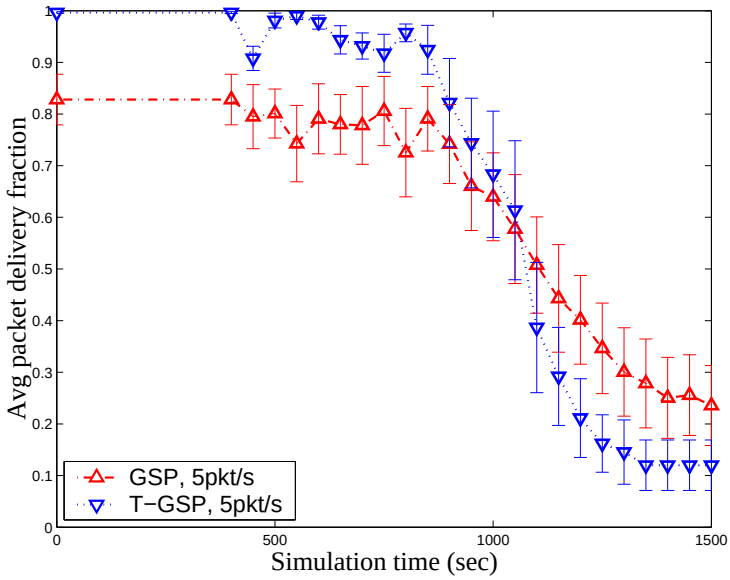

Figure 58: Average packet delivery fraction of GSP and T-GSP with long-lived CBR traffic (static network, traffic rate=5pkt/sec)

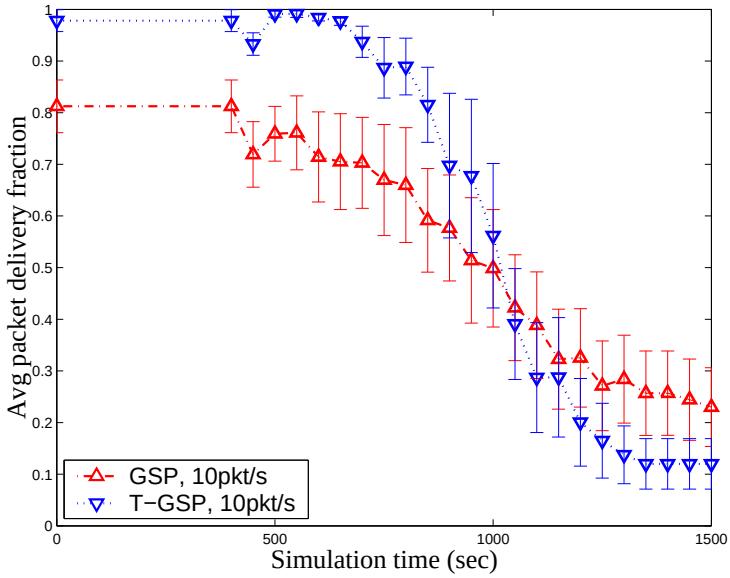

Figure 59: Average packet delivery fraction of GSP and T-GSP with long-lived CBR traffic (static network, traffic rate=10pkt/sec)

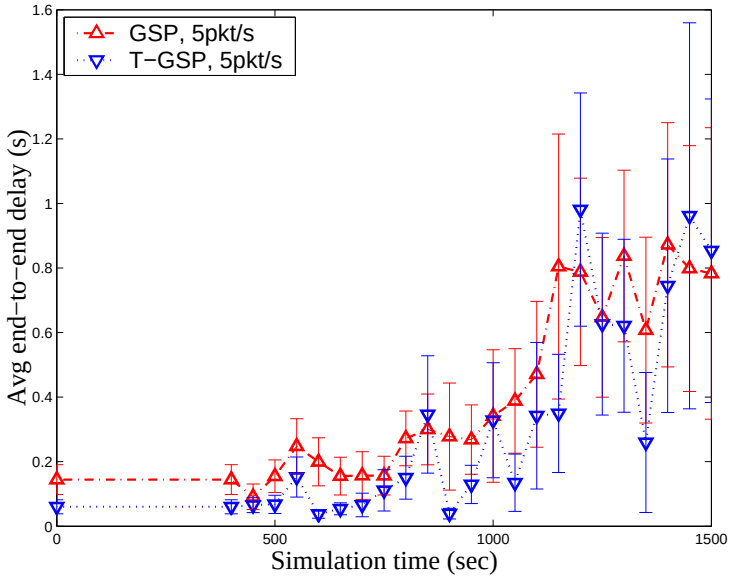

Figure 60: Average end-to-end delay of GSP and T-GSP with long-lived CBR traffic (pause time=0, traffic rate=5pkt/sec)

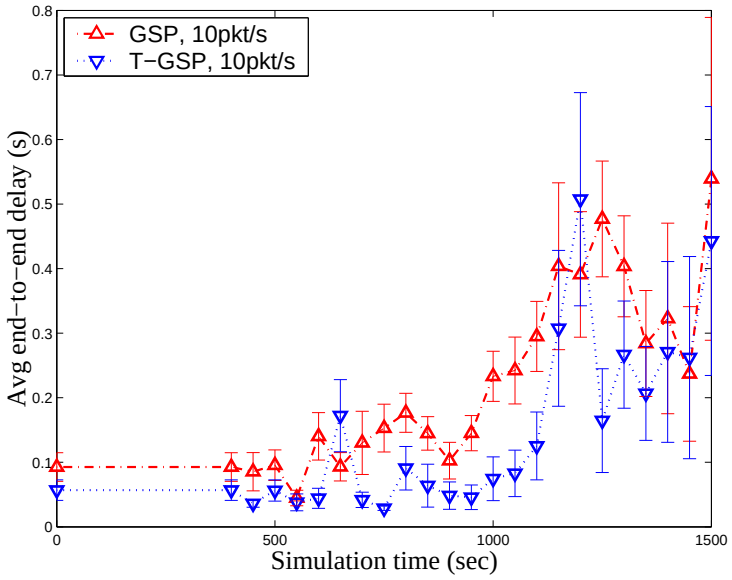

Figure 61: Average end-to-end delay of GSP and T-GSP with long-lived CBR traffic (pause time=0, traffic rate=10pkt/sec)

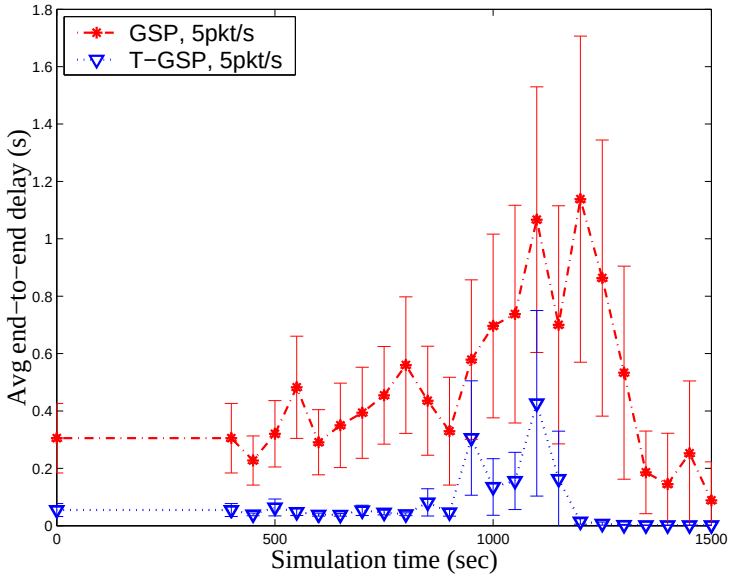

Figure 62: Average end-to-end delay of GSP and T-GSP with long-lived CBR traffic (static network, traffic rate=5pkt/sec)

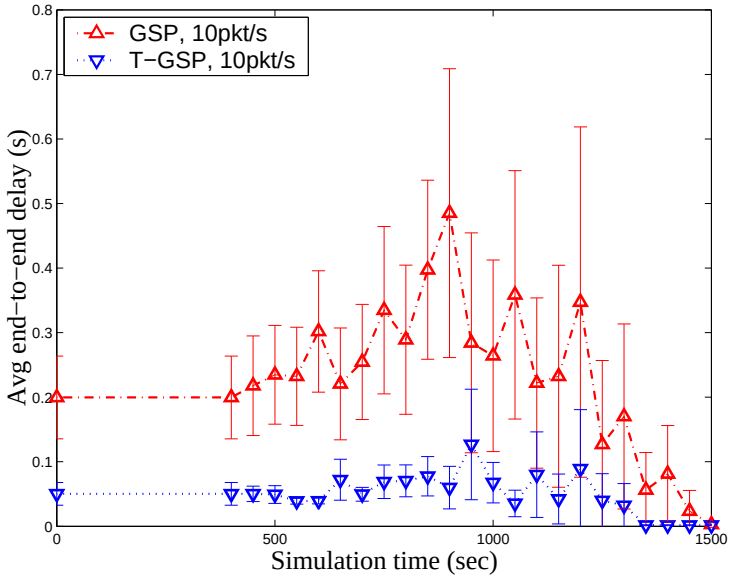

Figure 63: Average end-to-end delay of GSP and T-GSP with long-lived CBR traffic (static network, traffic rate=10pkt/sec)

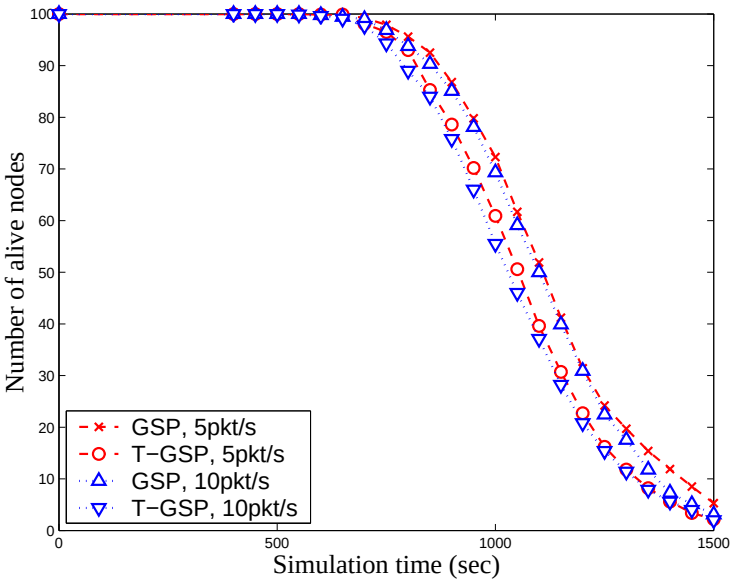

Figure 64: Network lifetime of GSP and T-GSP with different long-lived CBR traffic (pause time=0)

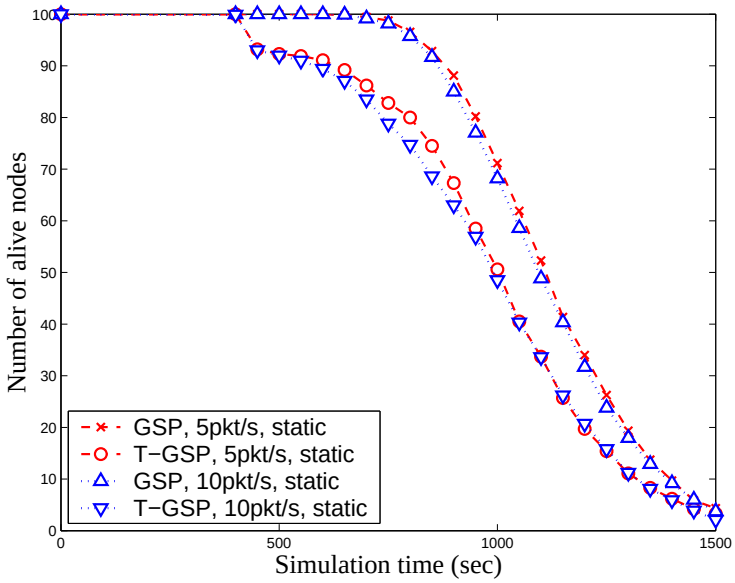

Figure 65: Network lifetime of GSP and T-GSP with different long-lived CBR traffic (static network)

the best values. We use asynchronous GSP and uniformly distributed gossip intervals with an average of 20 seconds. The gossip sleep probability is 0.7. Two mobility scenarios are considered, namely: (1) pause time of every node is 0 (i.e., constant motion) and (2) pause time of every node is infinity (i.e., static network). The simulation study of T-GSP is based on the same simulation model used in Section 4.1.

Two traffic loads were used, long-lived CBR and exponential on-off traffic based on UDP. Each packet carries 512 bytes of data payload, making the packet size 532 bytes including an IP header. The average packet rates are 5 and 10 packets/sec for long-lived CBR traffic and 10 and 20 packets/sec for on-off traffic. Both the busy and idle intervals of the on-off traffic follow an exponential distribution with a mean of 50 seconds.

The results are plotted with a 90% confidence interval. Fig. 56– 73 show the packet delivery fraction and end-to-end delay of GSP and T-GSP in case of long-lived CBR traffic with respect to the simulation time. Some curves, especially the end-to-end delay, show a high variability during the late part of the simulation. This is due to the network partition after more and more nodes

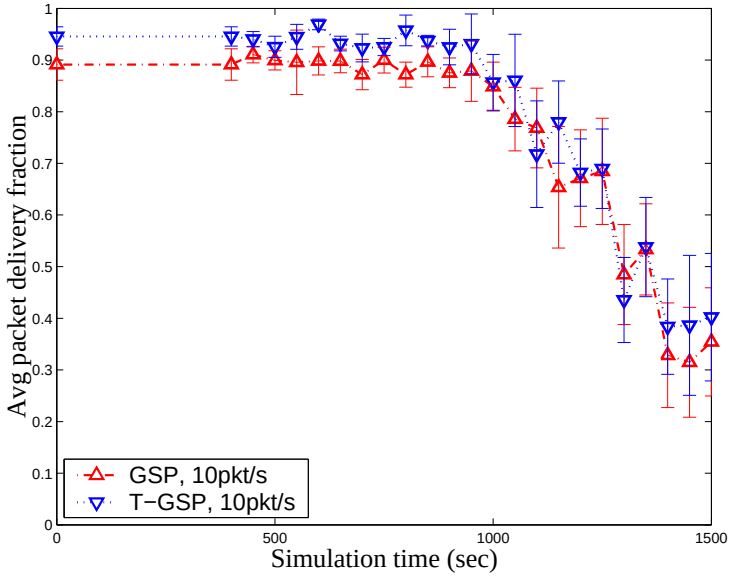

Figure 66: Average packet delivery fraction of GSP and T-GSP with exponential on-off traffic (pause time=0, traffic rate=10pkt/sec)

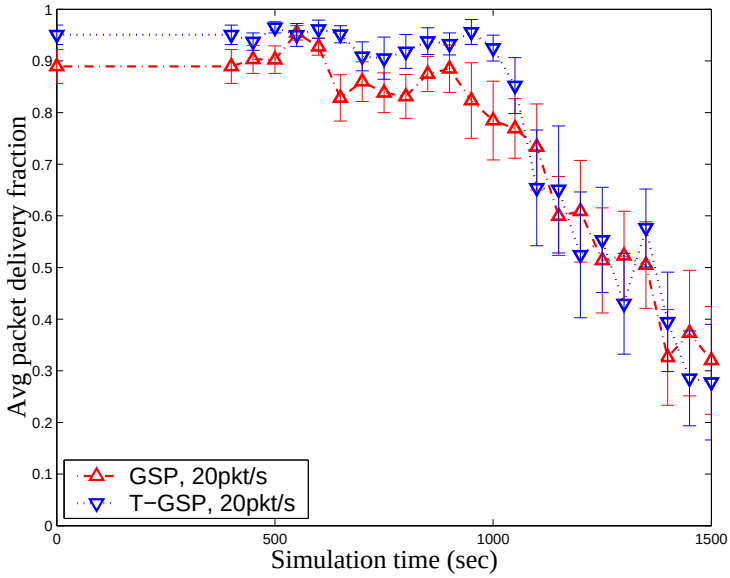

Figure 67: Average packet delivery fraction of GSP and T-GSP with exponential on-off traffic (pause time=0, traffic rate=20pkt/sec)

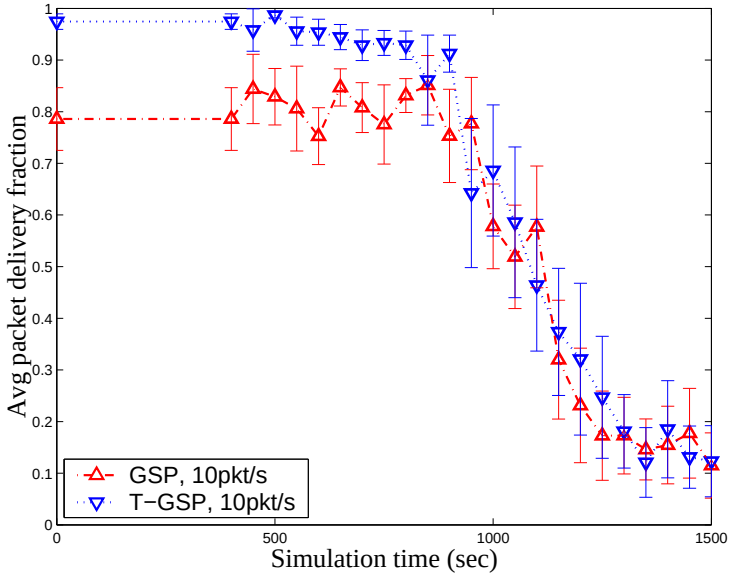

Figure 68: Average packet delivery fraction of GSP and T-GSP with exponential on-off traffic (static network, traffic rate=10pkt/sec)

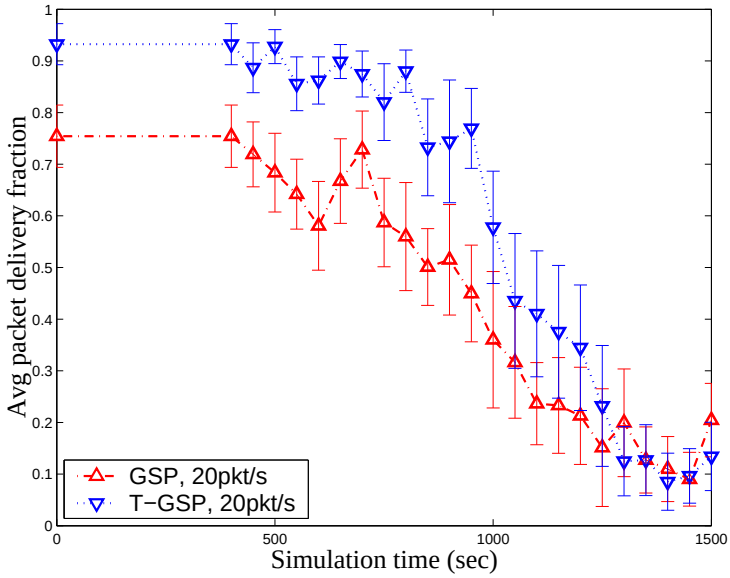

Figure 69: Average packet delivery fraction of GSP and T-GSP with exponential on-off traffic (static network, traffic rate=20pkt/sec)

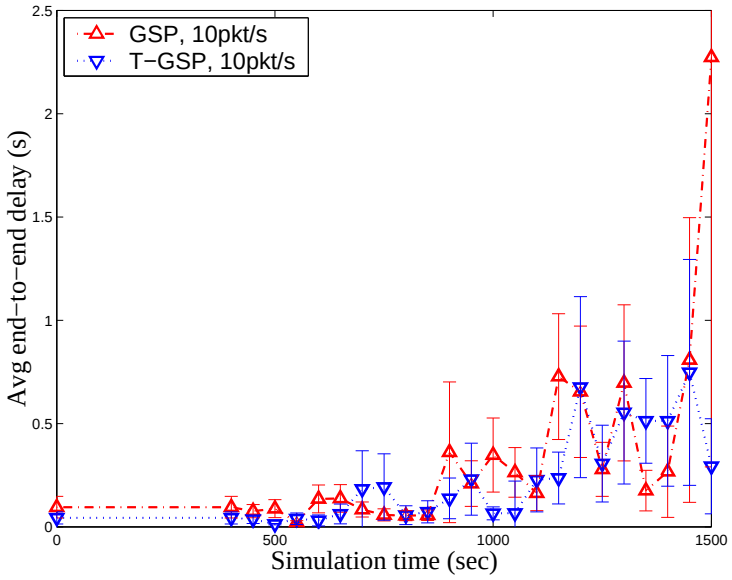

Figure 70: Average end-to-end delay of GSP and T-GSP with exponential on-off traffic (pause time=0, traffic rate=10pkt/sec)

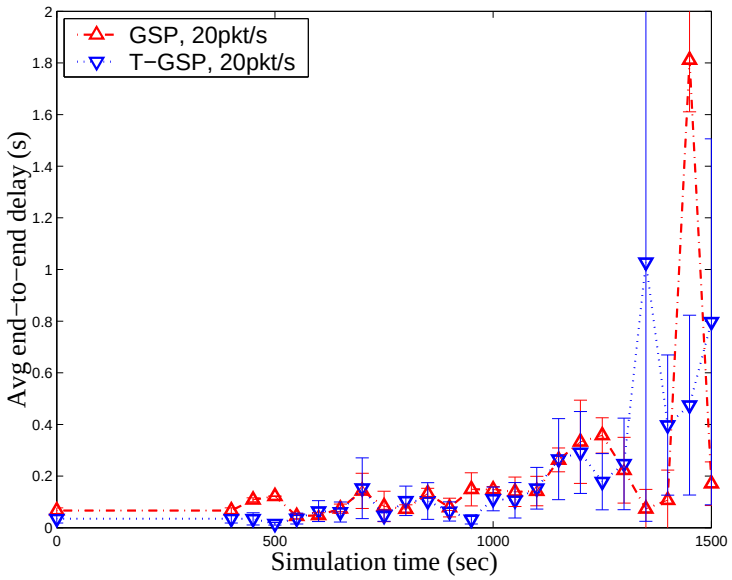

Figure 71: Average end-to-end delay of GSP and T-GSP with exponential on-off traffic (pause time=0, traffic rate=20pkt/sec)

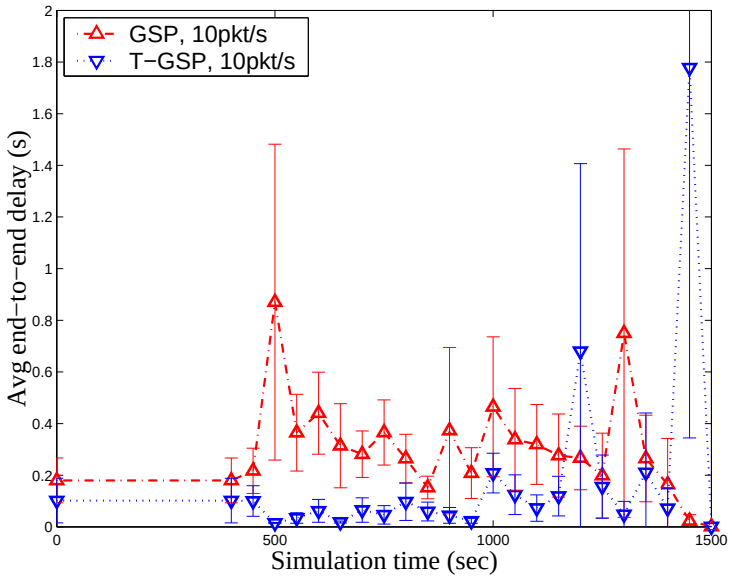

Figure 72: Average end-to-end delay of GSP and T-GSP with exponential on-off traffic (static network, traffic rate=10pkt/sec)

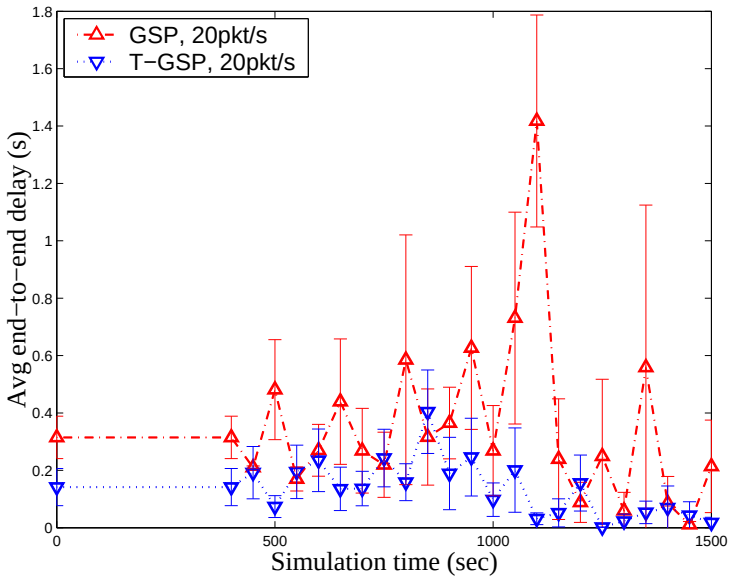

Figure 73: Average end-to-end delay of GSP and T-GSP with exponential on-off traffic (static network, traffic rate=20pkt/sec)

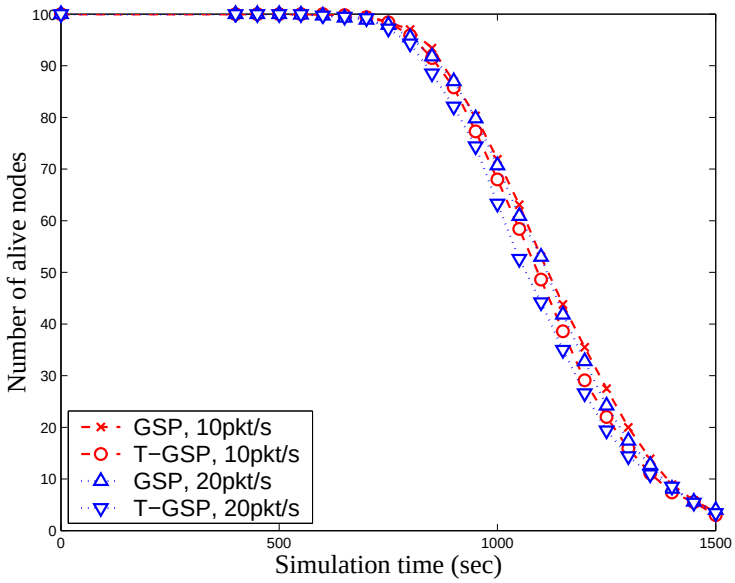

Figure 74: Network lifetime of GSP and T-GSP with different exponential on-off traffic (pause time=0)

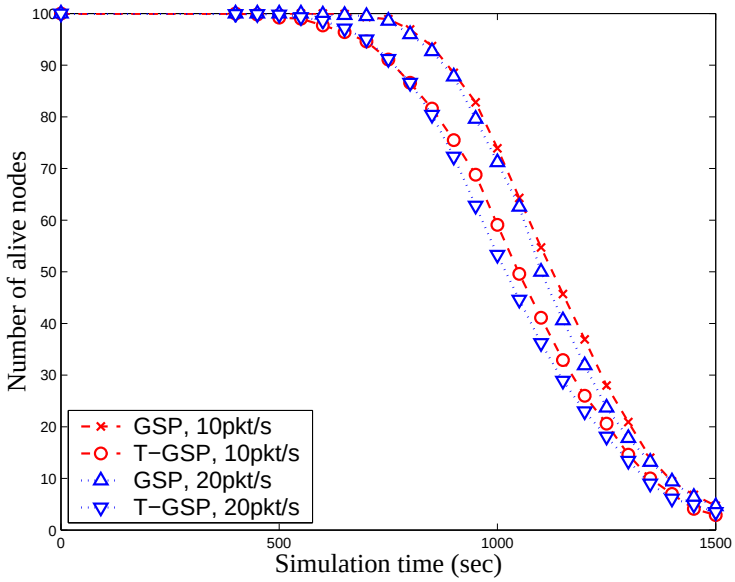

Figure 75: Network lifetime of GSP and T-GSP with different exponential on-off traffic (static network)

die. The traffic nodes may be connected or partitioned at different times, making the network performance change dramatically. In our work, we are more interested in the early part of the simulation, i.e., when the network is connected and network performance is high. In other words, we study the behavior of our protocols and network performance during the network lifetime.

From the figures, we can see that the packet delivery fraction and end-to-end delay are improved by the use of T-GSP. The improvement is more significant in case of a static network. The reason is that the capacity of a static network is lower than a mobile network with the same number of nodes [19] [10]. Therefore, a static network is more sensitive to the radio mode change introduced by GSP.

Fig. 64 and Fig. 65 show the network lifetime. The confidence intervals are not given since they are very small. One can see from the figure that the cost of T-GSP is more energy consumption incurred by the nodes with traffic to forward. We can see that the nodes die more quickly with T-GSP than GSP. This cost is more significant in a static network. In a static network with T-GSP, a path is used until some of the nodes die. So, nodes may be overused in this case. On the other hand, mobility can break an active path and make the energy consumption more evenly distributed among all the nodes.

Fig. 66– 75 show the three performance metrics for GSP and T-GSP in the case of exponential on-off UDP based traffic. We can see that similar results are achieved. Of course, since the on-off traffic incurs more path setup, its packet delivery fraction and end-to-end delay performance is slightly lower than the one of long-lived CBR traffic. It is worth noting that the lifetime of a static network is relatively better than the one with long-lived CBR traffic. Apparently, on-off traffic can reduce the overuse of a node due to the switching of busy and idle intervals. The next busy interval may use a different path from the previous one. The T-GSP results taken together show that by including local traffic information into the sleep mode protocol can improve the network performance.

5.4 SUMMARY

In this Chapter, we proposed several extensions of GSP to apply our work to a more realistic network. They adjust the gossip sleep probability according to some local information, such as local node density, remaining battery capacity, and traffic patterns. The extensions of GSP require either the local information from one-hop neighbors or an additional timer. We implemented a prototype for each extension in ns-2 to demonstrate how they work. The simulation results show that they can dynamically apply to different network scenarios and improve the performance of GSP in such scenarios.

In practice, we can either pick one or combine two or all of the extensions to deal with different network scenarios and meet different application requirements. The basic requirement to combine the extensions is to satisfy the sleep probability threshold to avoid the network partition, i.e., every node adopts a sleep probability smaller than the threshold required to connect the nodes in its area. We can achieve this by taking the minimum of the p values generated by different extensions, hence all individual extensions are satisfied. Of course, this may raise some concern due to the inherent conflicts among the extensions. For instance, when we combine B-GSP and T-GSP, a node forwarding traffic but with low remaining battery may be overused since T-GSP tends to generate a small or 0 sleep probability for this node while B-GSP usually generates a larger one. This situation can be improved by selecting proper parameters. For example, a small but non-zero p from T-GSP will allow the node to have a chance to sleep and the traffic may choose a different path. When it doesn't forward traffic, the node will mainly be controlled by B-GSP and sleep more. Similar thing happens when we combine A-GSP and T-GSP. A node carrying traffic in a dense area may be overused. A non-zero p from T-GSP can solve this problem. Of course, the best way to reduce the overuse problem is to employ B-GSP. The average gossip sleep probability of the entire network $\bar{p}$ in B-GSP can be configured as the sleep probability generated by A-GSP in each node.

Therefore, when we combine the all three extensions, the basic design rule is using A-GSP's output as B-GSP's input and B-GSP's output to design T-GSP. The layer structure should look like Fig. 76.

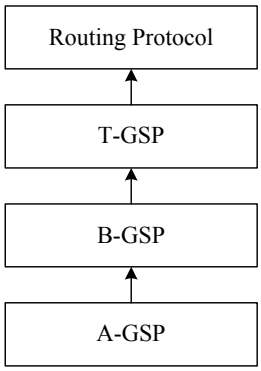

Figure 76: Layer structure to combine A-GSP, B-GSP and T-GSP

6.0 CONCLUSIONS

This dissertation has proposed a novel sleep management approach for wireless ad hoc including sensor networks. In this Chapter, the major contributions of this dissertation are summarized in Section 6.1 and possible future research directions are discussed in Section 6.2.

6.1 CONTRIBUTIONS

We have considered the problem of energy efficiency in wireless ad hoc and sensor networks. As we have reviewed in Chapter 2, existing energy conservation strategies in literature either do not take advantage of sleep modes to conserve energy more efficiently, or incur much overhead in terms of control message and computing complexity to schedule sleep modes and thus are not scalable. To fill the gap of these strategies, we have used the results of percolation theory, typically called *gossiping*, to manage the sleep of the nodes in a network. It is simple, scalable and capable of maintaining the network connectivity. Specifically, the major contributions of this dissertation include:

- We have developed a gossip-based sleep management protocol, i.e., GSP, for both static and mobile wireless ad hoc and sensor networks. The core idea is to employ probabilistic based sleep modes - essentially, tossing a coin to decide whether or not a node should sleep for the next interval. The strategy is very suitable to a large and low cost network. We have also examined the network connectivity and energy efficiency under this protocol by simulation and analysis.
- We have conducted a set of simulations to study the performance of GSP. The simulation re-

sults agree with the analytical results on network lifetime improvement. Furthermore, GSP was shown to only slightly affects the network performance in terms of packet delivery fraction and end-to-end delay. From the comprehensive simulation results, we have also explored the heuristics of p selection in certain network scenarios. In addition, we have discussed the possible impacts of the routing protocols with which GSP integrates.

- We have developed multiple extensions of GSP to improve the network performance and make the proposed protocol apply to different network scenarios. They adjust the gossip sleep probability according to some local information, such as local node density, remaining battery capacity, and ongoing traffic patterns, to adapt to a more realistic network. These extensions require only either the local information from one-hop neighbors or an additional timer, making the protocol keep its simplicity.

6.2 FUTURE RESEARCH

Our analytical and simulation results show that GSP can prolong the network lifetime without significantly sacrificing other network performance. However, further work is required to address various properties of GSP. Currently, the basic GSP requires a predetermined gossip sleep probability p. As mentioned in Section 3.1, there is no explicit expression of p and simulation has to be used to choose a suitable value of p for a given network scenario (density, mobility, etc.). It would be interesting and useful to find out a way to theoretically get it. In other words, in the future work we are interested in how to theoretically select the optimal value of p, that is the maximum value of the gossip sleep probability to avoid network partition in different random topologies, with and without mobility.

Another possible interesting topic is the traffic delay due to the network partition and how this can be overcome. As we can see, GSP cannot guarantee the connectivity from every node to every other node. Network connectivity is achieved at a certain high probability, i.e., network partition may still happen though with a very small probability, especially when the network size is large. Additionally, if we desire to prolong the lifetime as much as possible, we may use a sleep probability higher than the threshold to sacrifice the network connectivity. In this case, it would

be very useful to understand the relation between the possible long traffic delay and the low sleep probability. Of course, we also need the intermediate nodes to buffer the ongoing traffic waiting for the re-connection of the path.

In this dissertation, we have proposed several extensions of GSP to improve the performance and our simulation shows that they work quite well. However, there are still possible ways to refine them. For instance, to improve the T-GSP, we expect to find an adaptive traffic timer. A fixed timer doesn't fit all traffic patterns. In addition, T-GSP pays relatively more cost on the lifetime in a static network as shown in our simulations. Load balancing by rotating the active paths may improve the situation.

APPENDIX

PUBLICATIONS FROM THIS WORK

- X. Hou, D. Tipper, and S. Wu, "A Gossip-based Energy Conservation Protocol for Wireless Ad Hoc and Sensor Networks," *accepted by Journal of Network and Systems Management, special issue on Management of Wireless Ad Hoc Networks and Wireless Sensor Networks, 2006.*
- X. Hou, D. Tipper, and S. Wu, "Traffic-aware Gossip-based Energy Conservation for Wireless Ad Hoc and Sensor Network Routing," *IEEE Consumer Communications and Networking Conference (CCNC), Las Vegas, Nevada, 2006.*
- X. Hou and D. Tipper, "Adaptive Gossip-based Energy Conservation for Wireless Ad Hoc Routing," *IEEE Canadian Conference on Electrical and Computer Engineering (CCECE), Niagara Falls, Ontario, Canada, 2004.*
- X. Hou, D. Tipper, D. Yupho and J. Kabara, "GSP: Gossip-based Sleep Protocol for Energy Efficient Routing in Wireless Sensor Networks," *16th International Conference on Wireless Communications, Calgary, Alberta, Canada, 2004, (Wireless 2004).*
- X. Hou and D. Tipper, "Gossip-based Sleep Protocol (GSP) for Energy Efficient Routing in Wireless Ad Hoc Networks," *IEEE Wireless Communications and Networking Conference (WCNC), Atlanta, Georgia, 2004.*

BIBLIOGRAPHY

[1] A. Manjeshwar and D. P. Agrawal, "APTEEN: A Hybrid Protocol for Efficient Routing and Comprehensive Information Retrieval in Wireless Sensor Networks," in *IEEE International Parallel Distributed Processing Symposium*, 2002.

[2] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A Survey on Sensor Networks," *IEEE Communications Magazine*, Aug. 2002.

[3] B. Bellur and R. Ogier, "A Reliable, Efficient Topology Broadcast Algorithm for Dynamic Networks," in *Proceeding of IEEE INFOCOM*, 1999.

[4] R. V. Boppana and S. P. Konduru, "An Adaptive Distance Vector Routing Algorithm for Mobile Ad Hoc Networks," in *Proceeding of IEEE INFOCOM*, 2001.

[5] J. H. Chang and L. Tassiulas, "Routing for Maximum System Lifetime in Wireless Ad-hoc Networks," in *Proceeding of 37th Annual Allerton Conference on Communication, Control and Computing*, 1999.

[6] J. H. Chang and L. Tassiulas, "Energy Conserving Routing in Wireless Ad-hoc Networks," in *Proceeding of IEEE INFOCOM*, 2000.

[7] Y.-L. Chang and C.-C. Hsu, "Routing in Wireless/Mobile Ad -Hoc Networks via Dynamic Group Construction," *ACM Balzer Mobile Networks and Applications Journal*, vol. 5, pp. 27–37, 2000.

[8] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: An Energy-Efficient Coordination Algorithm for Topology Maintenance in Ad Hoc Wireless Networks," *Wireless Networks*, , no. 8, pp. 481–494, 2002.

[9] J. C. Chen, K. M. Sivalingam, P. Agrawal, and S. Kishore, "A Comparison of MAC Protocols for Wireless Local Networks Based on Battery Power Consumption," in *Proceeding of IEEE INFOCOM*, 1998.

[10] T. Chu and I. Nikolaidis, "Node Density and Connectivity Properties of the Random Waypoint Model," *Computer Communications*, , no. 27, pp. 914–922, 2004.

[11] T. Clausen, P. Jacquet, J. Laouiti, P. Minet, P. Muhlethaler, A. Qayyum, and L. Viennot, "Optimized Link State Routing Protocol," *IETF Internet Draft*.

[12] B. Das, R. Sivakumar, and V. Bharghavan, "Routing in Ad Hoc Networks Using a Spine," in *Proceeding of IEEE International Conference on Computer Communications and Networks(IC3N)*, 1997.

[13] Kevin Fall and Kannan Varadhan, *The ns Manual*, http://www-mash.cs.berkeley.edu/ns/, 2002.

[14] L. M. Feeney and M. Nilsson, "Investigating the Energy Consumption of a Wireless Network Interface in an Ad Hoc Networking Environment," in *Proceeding of IEEE INFOCOM*, 2001.

[15] Laura Marie Feeney, "An Energy Consumption Model for Performance Analysis of Routing Protocols for Mobile Ad Hoc Networks," *Mobile Networks and Applications*, vol. 6, no. 3, pp. 239–249, 2001.

[16] J.J. Garcia-Luna-Aceves and M. Spohn, "Source-Tree Routing in Wireless Networks," in *Proceeding of IEEE International Conference on Network Protocols*, 1999.

[17] T. Goff, N. B. Abu-ghazaleh, d. S. Phatak, and R. Kahvecioglu, "Preemptive Routing in Ad Hoc Networks," in *Proceeding Of ACM SIGMOBILE*, July 2001.

[18] G. Grimmett, *Percolation*, Springer-Verlag, 1989.

[19] M. Grossglauser and D. Tse, "Mobility Increases the Capacity of Ad-hoc Wireless Networks," in *Proceeding of IEEE INFOCOM*, 2001.

[20] Z. Haas, "A New Routing Protocol for Reconfigurable Wireless Networks," in *Proceeding Of IEEE International Conference On Universal Personal Communications*, 1997.

[21] Z. Haas and M. Pearlman, "The Performance of Query Control Schemes for the Zone Routing Protocol," in *Proceeding Of ACM SIGCOMM*, 1998.

[22] Z. J. Haas, J. Y. Halpern, and L. Li, "Gossip-based Ad Hoc Routing," in *Proceeding of IEEE INFOCOM*, 2002.

[23] S. Hedetniemi, S. Hedetniemi, and A. Liestman, "A survey of Gossiping and Broadcasting in Communication Networks," *Networks*, vol. 18, 1988.

[24] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-Efficient Communication Protocol for Wireless Microsensor Networks," in *IEEE Hawaii International Conference on System Sciences*, 2000.

[25] W. R. Heinzelman, J. Kulik, and H. Balakrishnan, "Adaptive Protocols for Information Dissemination Protocol for Wireless Sensor Networks," in *Proceeding Of ACM MOBICOM*, 1999.

[26] Y.-C. Hu and D. B. Johnson, "Implicit Source Routes for On-Demand Ad Hoc Network Routing," in *Proceeding Of ACM MOBIHOC*, 2001.

[27] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks," in *Proceeding Of ACM MOBICOM*, 2000.

[28] B. A. Iwata, C.-C. Chiang, G. Pei, M. Gerla, and T. W. Chen, "Scalable Routing Strategies for Ad Hoc Wireless Networks," *IEEE Journal on Selected Areas of Communications*, vol. 17, no. 8, Aug. 1999.

[29] D. Johnson, "Routing in Ad Hoc Networks of Mobile Hosts," in *Proc. IEEE Workshop on Mobile Computing Systems and Applications*, Dec. 1995.

[30] D. Johnson and D. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks," *Mobile Computing, T. Imielinski and H. Korth, eds.*, 1996.

[31] D. Johnson, D. Maltz, Y.-C. Hu, and J. G. Jetcheva, "The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR)," *IETF Internet Draft*.

[32] C. E. Jones, K. M. Sivalingam, P. Agrawal, and J. C. Chen, "A Survey of Energy Efficient Network Protocols for Wireless Networks," *Wireless Networks*, vol. 7, no. 4, pp. 343–358, Sept. 2001.

[33] J. Jubin and J. D. Tornow, "The DARPA Packet Radio Network Protocols," *Proc. IEEE*, vol. 75, no. 1, pp. 21–32, Jan. 1987.

[34] J. Widmer M. Mauve and H. Hartenstein, "A Survey on Position-Based Routing in Mobile Ad Hoc Networks," *IEEE Network*, Nov. 2001.

[35] J. Macker and S. Corson, "Mobile Ad Hoc Networks (MANET)," *IETF WG Charter., http://www.ietf.org/html.charters/manet-charter*, 1997.

[36] A. Manjeshwar and D. P. Agrawal, "TEEN: A Routing Protocol for Enhanced Efficiency in Wireless Sensor Networks," in *IEEE International Parallel Distributed Processing Symposium*, 2001.

[37] A. B. McDonald, *A Mobility-Based Framework for Adaptive Dynamic Cluster-Based Hybrid Routing in Wireless Ad Hoc Networks*, Ph.D. Dissertation, Univ. of Pittsburgh, 2000.

[38] A. B. McDonald and T. Znati, "A Mobility-Based Framework for Adaptive Clustering in Wireless Ad Hoc Networks," *IEEE Journal on Selected Areas of Communications*, vol. 17, no. 8, Aug. 1999.

[39] R. Meester and R. Roy, *Continuum Percolation*, Cambridge University Press, 1996.

[40] S. Murthy and J. J. Garcia-Luna-Aceves, "An efficient Routing Protocol for Wireless networks," *ACM Balzer Mobile Networks and Applications Journal*, 1996.

[41] C. M. Okino and M. G. Corr, "Best effort adaptive routing in statistically accurate sensor networks Neural Networks," in *Proceeding Of IJCNN*, 2002.

[42] M. Pearlman and Z. Haas, "Determining the Optimal Configuration for the Zone Routing Protocol," *IEEE Journal on Selected Areas of Communications*, vol. 17, no. 8, Aug. 1999.

[43] C. E. Perkins and P. Bhagwat, "Highly Dynamic Destination Sequenced Distance-Vector Routing (DSDV) for Mobile Computers," in *Proceeding Of ACM SIGCOMM*, Oct. 1994, pp. 234–244.

[44] C. E. Perkins and E. M. Royer, "Ad-hoc On-demand Distance Vector Routing," in *Proc. IEEE Workshop on Mobile Computing Systems and Applications*, 1999.

[45] C. E. Perkins, E. M. Royer, and S. R. Das, "Ad-hoc On-demand Distance Vector (AODV) Routing," *IETF Internet Draft*.

[46] G. J. Pottie and W. J. Kaiser, "Wireless Integrated Network Sensors," *Communications of the ACM*, pp. 51–58, 2000.

[47] V. Raghunathan, C. Schurgers, S. Park, and M. B. Srivastava, "Energy-Aware Wireless Microsensor Networks," *IEEE Signal Processing Magazine*, Mar. 2002.

[48] R. Ramanathan and M. Steenstrup, "Hierarchically-Organized, Multihop Mobile Wireless Networks for Quality-of-Service Support," *ACM Balzer Mobile Networks and Applications Journal*, vol. 3, pp. 101–119, 1998.

[49] S. Roy and J.J. Garcia-Luna-Aceves, "Using Minimal Source Trees for On-Demand Routing in Ad Hoc Networks," in *Proceeding of IEEE INFOCOM*, 2001.

[50] E. M. Royer and C.-K. Toh, "A Review of Current Routing Protocols for Ad Hoc Mobile Wireless Networks," *IEEE Personal Communications*, Apr. 1999.

[51] C. A. Santivanez, R. Ramanathan, and I. Stavrakakis, "Making Link-State Routing Scale for Ad Hoc Networks," in *Proceeding Of ACM MOBIHOC*, 2001.

[52] C. Schurgers and M. Srivastava, "Energy Efficient Routing in Wireless Sensor Networks," in *Proceeding of IEEE MILCOM*, 2001.

[53] C. Schurgers, V. Tsiatsis, S. Ganeriwal, and M. Srivastava, "Topology Management for Sensor Networks: Exploiting Latency and Density," in *Proceeding Of ACM MOBIHOC*, 2002.

[54] K. Scott and N. Bambos, "Routing and Channel Assignment for Low Power Transmission in PCS," in *Proceeding Of IEEE International Conference On Universal Personal Communications*, 1996.

[55] R. Shah and J. Rabaey, "Energy Aware Routing for Low Energy Ad Hoc Sensor Networks," in *Proceeding of IEEE WCNC*, 2002.

[56] S. Singh, M. Woo, and C. Raghavendra, "Power-Aware Routing in Mobile Ad Hoc Networks," in *Proceeding Of ACM MOBICOM*, 1998.

[57] R. Sivakumar, B. Das, and V. Bharghavan, "Spine Routing in Ad Hoc Networks," *ACM/Baltzer Cluster Computing Journal*, vol. 1, pp. 237–248, Nov. 1998.

[58] M. Spohn and J.J. Garcia-Luna-Aceves, "Neighborhood Aware Source Routing," in *Proceeding Of ACM MOBIHOC*, 2001.

[59] C.-K. Toh, "Maximum Battery Life Routing to Support Ubiquitous Mobile Computing in Wireless Ad Hoc Networks," *IEEE Communications Magazine*, June 2001.

[60] S. Udani and J. Smith, "Power Management in Mobile Computing (A Survey)," *Technical Report, http://www.cis.upenn.edu/˜udani/papers.html*, 1996.

[61] Y. Xu, J. Heildemann, and D. Estrin, "Geography-informed Energy Conservation for Ad Hoc Routing," in *Proceeding Of ACM SIGMOBILE*, 2001.

[62] Fan Ye, A. Chen, Songwu Lu, and Lixia Zhang, "A scalable solution to minimum cost forwarding in large sensor networks," in *Proceedings of the International Conference on Computer Communications and Networks*, 2001.

[63] J. Yoon, M. Liu, and B. Noble, "Random Waypoint Considered Harmful," in *Proceeding of IEEE INFOCOM*, 2003.

[64] R. Zheng and R. Kravets, "On-demand Power Management for Ad Hoc Networks," in *Proceeding of IEEE INFOCOM*, 2003.