2013

Improving sensor network lifetime with wireless charging technology

Zi Li
Iowa State University

Improving sensor network lifetime with wireless charging technology

by

Zi Li

A dissertation submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Major: Computer Science

Program of Study Committee:
Wensheng Zhang, Co-major Professor
Daji Qiao, Co-major Professor
Ruan Lu
Johnny S. Wong
Carl K. Chang

Iowa State University

Ames, Iowa

2013

TABLE OF CONTENTS

LIST OF TABLES vi

LIST OF FIGURES vii

ABSTRACT ix

CHAPTER 1. INTRODUCTION 1

1.1 Wireless Charging Technology in Sensor Networks 1

1.2 Problem Identification 2

1.3 Proposed Research 4

CHAPTER 2. JOINT ROUTING AND CHARGING SCHEME TO PROLONG SENSOR NETWORK LIFETIME 6

2.1 Introduction 6

2.1.1 Wireless Charging Technology 6

2.1.2 Literature Survey 7

2.1.3 Contributions 8

2.2 Preliminaries 8

2.3 J-RoC: A Joint Routing and Charging Scheme 9

2.3.1 Routing Cost 11

2.3.2 Charging Scheduling Algorithm 13

2.3.3 Performance Upper Bound 19

2.4 Design and Implementation 20

2.4.1 Hardware Component 21

2.4.2 Software Component 21

2.5 Experimental Study 23
2.5.1 Experimental Setup 23
2.5.2 Evaluation Results 24
2.6 Simulation Study 26
2.6.1 Simulation Setup 26
2.6.2 Simulation Results 27
2.7 Conclusions 31

CHAPTER 3. JOINT CHARGING AND RATE ALLOCATION FOR UTILITY MAXIMIZATION IN RECHARGEABLE SENSOR NETWORKS 33

3.1 Introduction 33
3.1.1 Motivations 33
3.1.2 Literature Survey 34
3.1.3 Contributions 37
3.2 System Model 37
3.2.1 Network Utility Model 38
3.2.2 Wireless Charging Model 39
3.2.3 Network Sustainability 39
3.3 Analytical Study 40
3.3.1 Problem Formulation 40
3.3.2 Approximate Algorithm 42
3.4 The proposed JCRA scheme 44
3.4.1 Charging Scheduling 46
3.4.2 Data Rate Allocation 46
3.5 Performance Evaluation 50
3.5.1 Simulation Setup 50
3.5.2 Simulation Results 52
3.6 Conclusions 56

CHAPTER 4. LIFETIME BALANCED DATA AGGREGATION IN DELAY-BOUNDED ENERGY-HETEROGENEOUS SENSOR NETWORKS 57

4.1 Introduction 57

4.1.1 Motivations 57

4.1.2 Literature Survey 58

4.1.3 Contributions 59

4.2 System Model and Problem Statement 59

4.2.1 System Model 59

4.2.2 Problem Statement 61

4.3 Design Overview 62

4.4 Analytical Study 63

4.4.1 Nodal Lifetime 63

4.4.2 Nodal Data Input/Output Rates 65

4.4.3 Lowerbound of a Subtree's Data Output Rate 67

4.4.4 Maximum Total Aggregation and Transmission Delay Allowed for Each Source-Sink Path 68

4.4.5 Performance Upperbound 70

4.5 Detailed Design 71

4.5.1 Initial Phase 71

4.5.2 Adaption Phase 73

4.6 Implementation and Evaluation 76

4.6.1 Implementation 76

4.6.2 Testbed Experiment Results 76

4.7 Conclusions 83

CHAPTER 5. JOINT AGGREGATION AND MAC DESIGN TO PROLONG SENSOR NETWORK LIFETIME 84

5.1 Introduction 84

5.1.1 Motivation 84

5.1.2 Literature Survey . . . 85
5.1.3 Contributions . . . 86
5.2 System Model and Problem Statement . . . 87
5.2.1 System Models . . . 87
5.2.2 Problem Statement . . . 89
5.3 The Proposed JAM Scheme . . . 90
5.3.1 Aggregator Module . . . 91
5.3.2 Output Rate Estimation Module (OREM) . . . 92
5.3.3 Lifetime Estimation Module (LEM) . . . 93
5.3.4 Intra-node Cross-layer Collaboration Module (ICCM) . . . 93
5.3.5 JAM Kernel . . . 94
5.3.6 Other Design Considerations . . . 97
5.4 JAM Implementation . . . 99
5.5 Performance Evaluation . . . 100
5.5.1 NS-2 Simulations . . . 100
5.5.2 Testbed Experiments . . . 104
5.6 Conclusions . . . 106
CHAPTER 6. CONCLUSIONS AND POSSIBLE FUTURE TOPICS . . . 107
6.1 Research Contributions . . . 107
6.2 Possible Future Research Topics . . . 108

LIST OF TABLES

2.1 Notations Used in the Joint Routing and Charging Problem 10

2.2 Simulation Parameters 27

3.1 Notation Summary 40

LIST OF FIGURES

1.1	An example of the wireless chargeable sensor network.	2
2.1	System Overview.	9
2.2	Overview of the proposed J-RoC scheme.	9
2.3	Computation of $\hat{e}_{i,j,t}$.	13
2.4	An example of calculating $c'_{i,t}$ values.	17
2.5	An example of the movement refinement.	18
2.6	Conceptual sketch of the software component.	22
2.7	Battery energy profile.	23
2.8	Experimental results.	25
2.9	Achieved network lifetime comparison with varying T_c and u.	29
2.10	Achieved network lifetime comparison with varying η, r_i and v.	30
3.1	System Overview.	38
3.2	Data rate determination overview in JCRA.	45
3.3	Topology used to describe JCRA details.	45
3.4	An example of JCRA initial phase.	47
3.5	Flow chart of data rate adaptation when node i acts as parent.	49
3.6	Historical solar power profile.	52
3.7	Performance comparison under different E_t.	52
3.8	Performance comparison under different τ.	53
3.9	Performance comparison under different I_{MC}.	55
3.10	Performance comparison under different network sizes.	55

3.11 Performance comparison under different percentage of near-trajectory nodes. 56

4.1 System model. 60
4.2 Design overview. 62
4.3 The sketch of RI-MAC protocol. 64
4.4 Flowchart of adjusting w_i when node i acts as a parent node. 74
4.5 Flowchart of adjusting w_i when node i acts as a child node. 75
4.6 Network topology in testbed experiments. 77
4.7 Performance comparison when all nodes generate data packets. 79
4.8 Performance comparison when only leaf nodes generate data packets. 80
4.9 Performance comparison under non-uniform initial nodal energy. 81
4.10 Trace of aggregation delays and nodal energy. 81
4.11 Impact of the degree of deviation in nodal energy levels. 82
4.12 Performance comparison under various data generation rates. 83

5.1 An RI-MAC like protocol but with a tunable Tr parameter. 88
5.2 JAM Overview. 90
5.3 Topology used to describe JAM details. 91
5.4 JAM kernel of node i. 95
5.5 JAM initialization example where the default Tr is 1 s and D is 15 s. 98
5.6 Implementation of JAM in TinyOS. 99
5.7 TelosB power meter kit used in JAM. 100
5.8 Simulation results under different end-to-end delay requirements. 101
5.9 Simulation results under different initial energy heterogeneity levels. 103
5.10 Simulation results under different network densities. 104
5.11 Network topology in testbed experiments. 104
5.12 Experiment results under different end-to-end delay requirements. 105
5.13 Changing traces of nodal lifetime, FAD, and Tr values. 106

ABSTRACT

Wireless sensor networks have been widely employed in a broad range of applications. As powered by small batteries, the scarce energy supply has constrained sensor node lifetime. The emerging wireless charging technology is a promising alternative to address such energy constraint problem in sensor networks. Comparing to existing approaches, this technology can replenish energy in a more controllable manner and does not require accurate location of or physical alignment to sensor nodes. However, little work has been reported on exploiting wireless charging to improve sensor network lifetime. In this dissertation, we study the network lifetime elongation problem in wireless chargeable sensor networks (WCSNs). Specifically, the study is conducted in four directions. First of all, with given sensing quality requirement, we study how to maximize the network lifetime. Secondly, with given network lifetime requirement, we study how to maximize the sensing quality. Thirdly, we study the network lifetime elongation in energy heterogeneous WCSNs by designing a lifetime balanced aggregation protocol. Fourthly, we jointly optimize the aggregation and MAC behaviors to further improve the WCSN network lifetime.

CHAPTER 1. INTRODUCTION

1.1 Wireless Charging Technology in Sensor Networks

Wireless sensor networks consist of spatially distributed sensor nodes to monitor physical or environmental conditions, such as temperature, sound, vibration, pressure and motion of pollutants, and to cooperatively pass their data through the network to a base station (BS). They have been widely employed in a broad range of applications related to environmental monitoring [1, 2], military [3], health care [4], national security [5] and so on. Many of these applications require long-term operation of the deployed sensor networks to reduce redeployment costs. However, sensor nodes are usually powered by small batteries and the scarce energy supply has constrained their lifetime. This has been a long-lasting, fundamental problem faced by sensor networks. To resolve this problem, various approaches like energy conservation [6, 7], ambient energy harvesting [8–11], incremental deployment, and battery replacement [12, 13] have been proposed. However, energy conservation schemes can only slow down energy consumption but not compensate energy depletion. Harvesting environmental energy, such as solar [8, 9], wind [10] and vibration [11], is subject to their availability which is often uncontrollable by people. The incremental deployment approach may not be environmentally friendly because deserted sensor nodes can pollute the environment. The battery or node replacement approach is applicable only for scenarios that sensor nodes are accessible by people or sophisticated robots that can locate and physically touch the sensor nodes.

The newly emerging wireless charging technology [14, 15] provides a promising alternative to address the energy constraint problem in sensor networks. Different from energy harvesting, wireless charging technology, together with more and more mature and inexpensive mobile robots, creates a controllable and perpetual energy source, with which power can be replenished proactively to meet application requirements rather than passively adapted to the availability of environmental resources.

Comparing with sensor node or battery replacement approaches, it allows a mobile charger to transfer energy to sensor nodes wirelessly without requiring accurate localization of sensor nodes or strict alignment between the charger and sensor nodes. Figure 1.1 shows an example of the wireless chargeable sensor network (WCSN) which is the integration of a sensor network and a mobile charger (MC). The MC is a mobile robot carrying a wireless charger. In this figure, sensor nodes collaboratively route the sensory data to the base station while the mobile charger moves along the trajectory and transfers energy to nodes nearby.

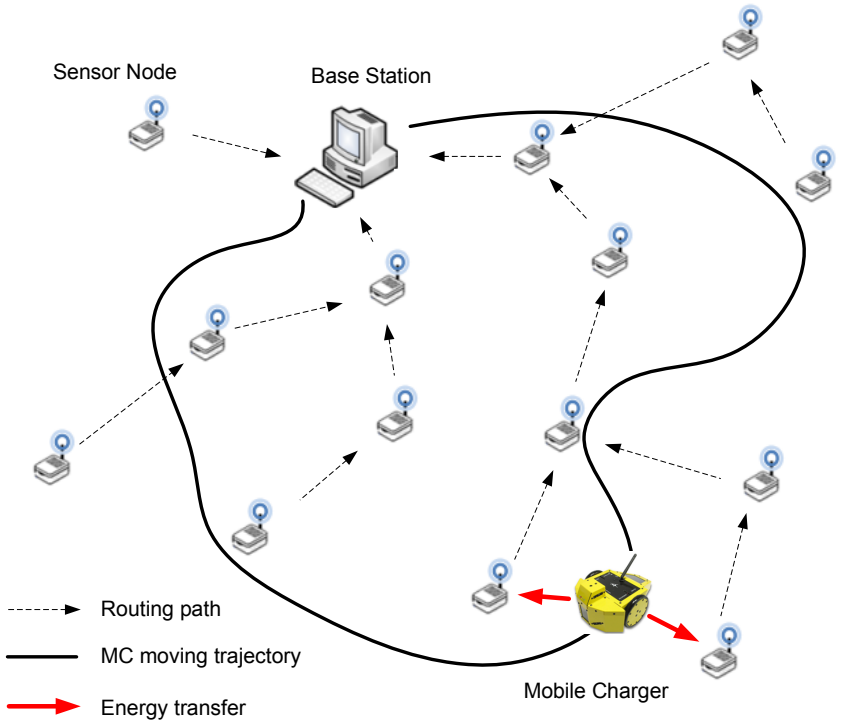


Figure 1.1 An example of the wireless chargeable sensor network.

1.2 Problem Identification

Introducing wireless charging technology into sensor networks adds a new dimension to network protocol design. The juncture point between wireless charging and network protocols is the network energy distribution. The former determines how the external energy flows into the network while the latter determines how the replenished energy is consumed. Ideally, these two parts shall be optimized

in a joint manner so that the network lifetime, e.g., the minimal nodal lifetime in the network, can be maximized. In this dissertation, the following topics will be discussed:

- First of all, as the routing strategy and energy charging scheduling both have significant impacts on nodal energy distribution and hence network lifetime, the study of joint routing and charging is a fundamental problem. By assuming every sensor node could be charged, it is of paramount importance to design schemes that can (i) make routing decision for individual nodes and (ii) schedule the charging sequence for the MC, so as to maximize the network lifetime. Such schemes should adapt to the dynamism in charging capabilities, network workload, link conditions, etc., and should have low operational overhead.

- Secondly, as the ambient energy is free and green, it is naturally an attractive idea to explore the possibility of building a sustainable sensor network, i.e., a network with eternal lifetime, by utilizing both the wireless charging and ambient energy harvesting technologies. This is also of critical importance considering the reality that, in some deployment fields, the MC may only move along some pre-determined trajectories and charge nodes near the trajectories due to the limited wireless power transfer range. How to maximize the collected sensory data utility under the sustainability requirement is an interesting problem. Clearly, an ideal solution shall determine the charging activities and nodal sensory data generation rates collaboratively while taking the ambient energy temporal-spatial varying nature into account.

- Thirdly, unequal opportunities of being charged and the temporal-spatial varying ambient energy harvesting rate, together with other factors like uneven workload, different battery qualities, etc., may result in heterogeneous nodal energy distribution. Moreover, data aggregation has been common in sensor networks. Hence, it is of practical importance to design data aggregation schemes in sensor networks with highly heterogeneous energy supply, so as to maximize the network lifetime.

- Finally, different from data aggregation schemes which affect the network lifetime from the network traffic perspective, MAC protocols affect the networks lifetime via the communication overhead distribution between the sender and receiver node pairs. As data aggregation and MAC

protocols affect the network lifetime from two distinct perspectives, a joint design may better handle the heterogeneous energy supply and further improve the network lifetime. Hence, a cross-layer optimization shall be conducted.

1.3 Proposed Research

In this dissertation, we study the network lifetime elongation problem in WCSNs. We aim at designing some deployable network protocols and the MC scheduling algorithms to fully exploit the strength of the charging technology and hence to maximize the network lifetime. The following research topics are included:

- *Joint Routing and Charging Scheme to Prolong Sensor Network Lifetime.*

  We study the impact of wireless charging technology in data collection applications and propose a practical and efficient Joint Routing and Charging scheme named J-RoC to prolong the network lifetime. To evaluate its performance, we conduct experiments in a testbed consisting of TelosB sensor nodes and Powercast [14] wireless charger plus Garcia [16] robots. Evaluation results demonstrate that J-RoC significantly elongates the network lifetime compared to existing wireless charging based schemes.

- *Joint Charging and Rate Allocation for Utility Maximization in Sustainable Sensor Networks.*

  Maximizing the sensed data utility for sustainable sensor networks [17, 18] is critically important when the networks are deployed for monitoring applications. To address this issue, we build the sustainable network with wireless charging and ambient energy harvesting technologies and propose a distributed scheme called JCRA to jointly schedule the charging activities and data rate allocation while considering the ambient energy temporal-spatial varying nature. NS-2 simulation results demonstrate that the network utility achieved by JCRA approaches the theoretical optimal solution under various network settings.

- *Lifetime Balanced Data Aggregation in Delay-Bounded Energy-Heterogeneous Sensor Networks.*

  To extend the lifetime of a network with highly heterogeneous energy supply, we first design a data aggregation scheme with a given routing topology. Particularly, based on the idea of bal-

ancing nodal lifetime through lifetime-aware arrangement of data aggregation holding time, we propose LBA, a delay-bounded Lifetime-Balanced data Aggregation scheme. Extensive experimental studies on a sensor network testbed show that LBA yields longer network lifetime than other data aggregation schemes and approaches the theoretical upperbound performance.

- *Joint Aggregation and MAC Design to Prolong Energy-Heterogeneous Sensor Network Lifetime.* Based on the LBA design and bounded end-to-end delay requirement, we conduct a further study to design a joint aggregation and MAC scheme, call JAM, to balance the nodal lifetime via co-adapting the data aggregation holding time and MAC protocol sleeping interval in energy-heterogeneous networks. Extensive ns-2 simulation and TinyOS experiment results demonstrate the effectiveness of JAM in prolonging the network lifetime compared with LBA.

The rest of the dissertation is organized as follows. In Chapter 2, we propose our J-RoC scheme. Chapter 3 present our JCRA design. The LBA and JAM schemes are presented in Chapters 4 and 5, respectively. Chapter 6 concludes this dissertation with a summary of the main contributions and discusses possible future research topics.

CHAPTER 2. JOINT ROUTING AND CHARGING SCHEME TO PROLONG SENSOR NETWORK LIFETIME

2.1 Introduction

Extending the sensor network lifetime for long-term operation is a long-lasting and fundamental problem. To address this issue, harvesting the ambient energy such as solar [8], wind [10] and vibration [11] has recently been proposed, and has attracted a lot of research [19, 20]. However, a limitation of the energy harvesting-based approach is that it is subject to the availability of the ambient energy, which is uncontrollable.

2.1.1 Wireless Charging Technology

Complementary to harvesting the ambient energy, the emerging wireless charging technology creates a perpetual power source to provide power-over-distance, one-to-many charging, and controllable wireless power. Particularly, employing two strongly coupled magnetic resonant objects, Kurs *et al.* [15] exploit the resonant magnetic technique to transfer energy from one storage device to another without any plugs or wires. The reported experiment demonstrated a wireless illumination of a 60 W light bulb from 2 meters away and achieved a 40% energy transfer efficiency. Zhang *et al.* [21] apply this technique to replenish battery energy in medical sensors and implantable devices in health care industry. Products from Powercast [14] carry out wireless charging by leveraging the electromagnetic radiation technique, with which energy transmitters broadcast the RF energy and receivers capture the energy and convert it to DC. Applications of the electromagnetic radiation technique for wireless charging have been reported in [22,23]. As more and more applications of wireless charging technology have been envisioned, the Wireless Power Consortium [24] has recently been established to start the efforts of setting an international standard for interoperable wireless charging.

2.1.2 Literature Survey

The application of wireless charging technology to sensor networks is still in its infancy stage. Peng *et al.* [22] recently study the feasibility of using the wireless charging technology to prolong the sensor network lifetime in a prototype system. The key idea is to dispatch a mobile robot to move around the network and charge energy to a selected set of lifetime-bottleneck sensor nodes. As the protocols run by sensor nodes should be simple and localized, the system employs two well-known routing protocols, i.e., energy-balanced routing and energy-minimum routing, both unaware of wireless charging activities. The charging strategy adopted by the system is simply to charge nodes with the lowest residual nodal lifetime. Hence, the wireless charger only passively makes up for the energy deficiency in the bottleneck nodes caused by the routing activities; that is, the charging activities are passively affected by the routing activities. This may result in the following undesired consequences. If energy-minimum routing is used, nodes on the intersection of multiple energy-minimum routes may be overused even though the charger keeps charging them. When the energy consumption rates of these nodes exceed the charging capability, they deplete their energy quickly and the extension in the network lifetime is limited. Alternatively, if energy-balanced routing is used, the overall energy consumption in the network is increased as routes with longer length (and hence higher energy consumption) are used to bypass low-energy nodes which are on shorter and more energy-efficient routes. Hence, the energy replenished into the network may not be utilized efficiently.

In another recently reported effort, Shi *et al.* [25] conduct theoretical study on efficient usage of the wireless charging technology in sensor networks. Based on the assumptions that the wireless charging capability is high enough to maintain an eternal network lifetime, the traffic pattern is fixed and the communication channels are perfect, they formulate and solve the problem of maximizing the ratio of the wireless charging vehicle's vacation time over each renewable energy cycle. Their solution is a static, centralized joint routing and charging algorithm. Hence, it may not be practical when the charging capability is constrained, the link qualities are imperfect and time-varying or the nodal energy consumption rates are heterogeneous and time-varying.

2.1.3 Contributions

To maximize the network lifetime under the constraint of limited charging capability, dynamic and imperfect communication environment, and heterogeneous node attributes, we propose *J-RoC*, a practical joint routing and charging scheme, in this study. J-RoC aims to employ energy-balanced routing and energy-minimum routing in a balanced way to exploit their strengths while avoiding or mitigating the problems caused by using only one of them. For this purpose, J-RoC requires periodical information exchanges between sensor nodes and the charger. Based on the exchanges, the charger keeps track of the global energy status of the network, schedules its charging activities accordingly, and disseminates the charging schedule to the network. Meanwhile, sensor nodes use a carefully designed *charging-aware* routing metric to estimate their routing costs and make routing decisions; this way, sensor nodes are guided to balance between energy-balanced routing and energy-minimum routing while the protocols run by them remain simple and localized.

We have implemented J-RoC and experimented in a small-scale TelosB sensor network testbed. In addition, extensive simulations have been conducted to study the performance of the J-RoC scheme in large-scale networks. Evaluation results show that J-RoC yields significantly longer network lifetime than existing solutions.

2.2 Preliminaries

As illustrated in Figure 2.1, we consider a system composed of three main components: a mobile charger (MC) that is a mobile robot carrying a wireless power charger, a network of sensor nodes each equipped with a wireless power receiver, and a base station (BS) that monitors the energy status of the network and directs the MC to charge sensor nodes.

The system works as follows. Each sensor node generates sensory data and sends the data hop-by-hop to the sink periodically. It also measures its local energy level, monitors the channel conditions, estimates its energy consumption rate, and reports these information along with the data packet generation rate to the BS. Based on the collected information, the BS schedules future charging activities, and commands the MC via a long range radio to execute the schedule. The MC then travels around the deployment field to charge sensor nodes. The BS also disseminates the schedule to sensor nodes,

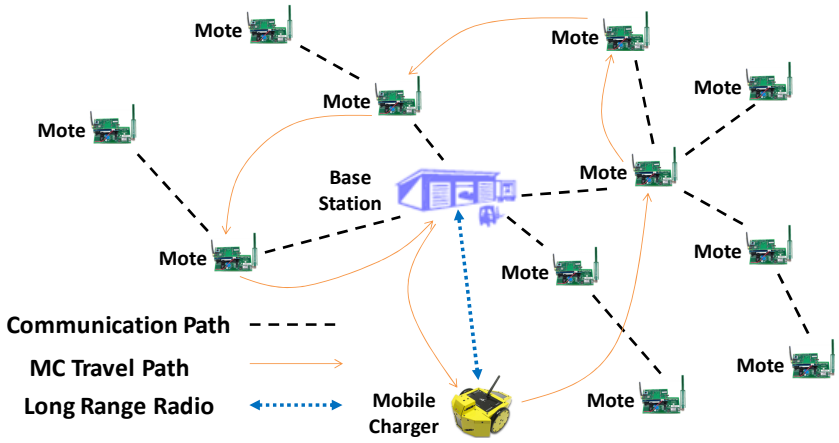


Figure 2.1 System Overview.

which may be used in routing path construction. We assume the MC's energy can be replenished at the BS and thus the energy for moving and charging is unlimited.

Notations used in this study are listed in Table 2.1.

2.3 J-RoC: A Joint Routing and Charging Scheme

In this section, we present J-RoC – a Joint Routing and Charging scheme. As shown in Figure 2.2, J-RoC works through periodical interactions between the sensor nodes, the base station (BS) and the mobile charger (MC).

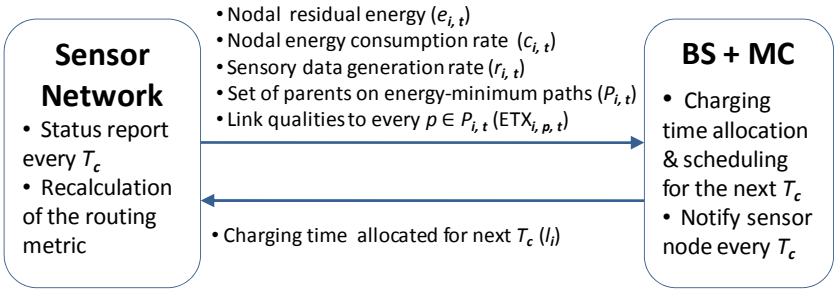


Figure 2.2 Overview of the proposed J-RoC scheme.

Every T_c time, the BS determines a charging schedule (i.e., charging time l_i for each sensor node i) for the next T_c interval. As detailed in Section 2.3.2, the schedule is decided based on the following information reported by each node: its energy consumption rate, residual energy level, data packet

Table 2.1 Notations Used in the Joint Routing and Charging Problem

notation	meaning
E_s	battery capacity of a sensor node
e_{tx}	energy consumed for transmitting a packet
e_{rx}	energy consumed for receiving a packet
Λ_c	energy consumed for the MC's charging operation
η	MC's charging efficiency
v	MC's moving speed
T_c	charging activity scheduling interval
α	charging guiding coefficient
l_i	charging time allocated to node i in one T_c interval
$r_{i,t}$	future sensory data packet generation rate of node i estimated at time t
$\varphi_{i,t}$	amount of time that node i has been charged in the current T_c interval at time t
$h_{i,t}$	charging rate of node i at time t
$e_{i,t}$	residual energy of node i at time t
$c_{i,t}$	energy consumption rate of node i at time t
$c'_{i,t}$	energy consumption rate of node i at time t in transceiving via the energy-minimum routing paths
$\hat{c}_{i,t}$	future energy consumption rate of node i estimated at time t
$P_{i,t}$	set of node i's parents on the energy-minimum path at time t
$ETX_{i,j,t}$	expected number of transmissions needed to send a packet successfully from node i to j at time t
ρ_i	percentage of charging energy that should be allocated to node i in one T_c interval

generation rate, set of parents on the energy-minimum paths to the sink, and the qualities of links to each parent. The BS disseminates the schedule to nodes and commands the MC to execute it. To reduce the control message overhead incurred by the periodical interactions between the BS and the network, the interaction interval can be configured to be much larger than the sensor data report interval. This will not compromise the system performance much, because the status of the network likely will not change significantly until a relatively large amount of data have been transmitted and received. For example, the data report interval is 2.5 seconds in our testbed experiments, and we set the interaction interval to be one hour. Evaluation results in Section 2.5 show that such a configuration performs well and yields a network lifetime that is reasonably close to the upper bound.

The Collection Tree Protocol (CTP) [26] is used as the routing protocol in J-RoC to report sensory data and nodal status to the BS. CTP is the default routing protocol in TinyOS 2.x. It designates a

node in the network as the sink node. All other nodes recursively form routing trees rooted at the sink. Nodes periodically broadcast beacons which serve two purposes. Firstly, they contain a field that the link estimator component of TinyOS uses to estimate the expected number of transmissions needed to send a packet successfully (ETX) to a node's neighbor, which roughly reflects the reciprocal of the packet reception ratio ($\frac{1}{PRR}$) over the link. Secondly, nodes embed in these beacons an estimate of the total cost (zero for the sink node and ∞ for others, initially) of routing a data packet to the sink from them. Non-sink nodes then collect the advertised routing costs from their neighbors, add their own one-hop routing costs, and select the neighbors with the lowest total routing costs as their parents. Since the beacons are broadcasted periodically, nodes can dynamically change their parents as routing costs fluctuate.

In J-RoC, each sensor node embeds two types of routing costs in CTP beacons. One contains the total cost of routing a packet to the sink along a charging-aware path. The other one contains the cost of delivering a packet along the energy-minimum path. Here, the energy-minimum path is defined as the path with the minimum total energy consumption in delivering a packet from a source to a destination. Link quality has been considered in estimating the energy consumption. In J-RoC, all data packets are routed via the aforementioned charging-aware paths to the sink. Note that the paths are different from the conventional energy-balanced or energy-minimum ones; as to be elaborated, the selection of the paths considers simultaneously the effects of energy charging, energy balancing and energy efficiency.

2.3.1 Routing Cost

After receiving the costs from neighbor nodes, sensor node i calculates its *energy-minimum* routing cost ($\mathcal{C}'_i$) as follows:

$$\mathcal{C}'_i = \min_{j \in N_i} \left\{ \mathcal{C}'_j + ETX_{i,j,t} \right\}, \tag{2.1}$$

where N_i is the set of node i's neighbor nodes, $\mathcal{C}'_j$ is the routing cost of node j and $ETX_{i,j,t}$ represents the expected number of transmissions needed to send a packet successfully over link (i, j). Hence, Equation (2.1) computes the minimum number of transmissions needed to deliver a packet from i to the sink successfully. Note that when links are in perfect condition, e.g., $ETX_{i,j,t} = 1$ for any i and j, the energy-minimum path becomes the shortest path.

The *charging-aware* routing cost at node i ($\mathcal{C}_i$) is computed as follows:

$$\mathcal{C}_i = \min_{j \in N_i} \left\{ \mathcal{C}_j + u^{1-\frac{\hat{e}_{i,j,t}}{E_s}} \right\}, \tag{2.2}$$

where $\mathcal{C}_i$ is the routing cost of node j, E_s is the battery capacity of a sensor node, and $\hat{e}_{i,j,t}$ in routing metric $u^{1-\frac{\hat{e}_{i,j,t}}{E_s}}$ is computed as

$$e_{i,t} + (l_i - \varphi_{i,t})\Lambda_c \eta - t_r * c_{i,\hat{p}_{i,t},t} * \frac{ETX_{i,j,t}}{ETX_{i,\hat{p}_{i,t},t}}. \tag{2.3}$$

In Equation (2.3), $\varphi_{i,t}$ denotes how long node i has been charged in the current T_c interval, t_r represents the remaining time in the current T_c interval and $\hat{p}_{i,t}$ denotes the parent of node i on the charging-aware path at time t. The term $c_{i,\hat{p}_{i,t},t} * \frac{ETX_{i,j,t}}{ETX_{i,\hat{p}_{i,t},t}}$ estimates the nodal energy consumption rate if i switches its parent from $\hat{p}_{i,t}$ to j. As the nodal energy consumption rate $c_{i,t}$ is measured when $\hat{p}_{i,t}$ is i's parent, we abbreviate $c_{i,\hat{p}_{i,t},t}$ to $c_{i,t}$ in the following sections.

The purpose of using this routing cost is to balance the energy consumption in the possibly lossy wireless environment among sensor nodes in the presence of energy charging. If there is no energy charging, a well-known energy-balanced routing metric [27] is

$$u^{1-\frac{e_{i,t}}{E_s}}. \tag{2.4}$$

Though the energy-balanced routing extends the network lifetime, different approaches should be adopted when energy charging is available. With energy charging, as much as possible energy should be replenished into nodes on the energy-minimum paths, so that these nodes can live longer and allow others to use them for packet routing, which can improve the energy utilization efficiency and hence prolong the network lifetime. However, as charging takes long time to be accomplished, nodes selected to be charged may not often maintain a high residual energy level, and therefore, energy-minimum paths may not often be chosen by other nodes to route their packets if Equation (2.4) is used to compute the routing cost. Furthermore, in some environments, particularly in the 2.4 GHz frequency band, links could be highly lossy [28]. Without the knowledge of the link quality, lots of energy may be

wasted on packet retransmissions over lossy links.

Our proposed routing metric addresses the above problems by factoring in the effects of charging that has been planed but not executed yet to estimate the routing cost, as well as the real-time link quality. Specifically, $\hat{e}_{i,j,t}$ estimates node i's residual energy at the end of the current T_c interval when it selects node j as parent, based on the knowledge of the charging schedule and link quality as in Equation (2.3). Then, $\hat{e}_{i,j,t}$ instead of $e_{i,t}$ is used in the routing metric as in Equation (2.2). Hence, nodes are led to choose paths to balance their residual energy at the end of the current T_c.

Figure 2.3 demonstrates how $\hat{e}_{i,j,t}$ is estimated at time $t = t_{curr}$. Note that, in this example, the energy consumption rate of node i is assumed to be constant from the current time to the end of the T_c interval to simplify the estimation. Specifically, in Figure 2.3, $c_{i,t_{curr}}, e_{i,t_{curr}}, h_{i,t_{curr}}$ are the energy consumption rate, nodal residual energy and charging rate at time t_{curr}, respectively. $\varphi_{i,t_{curr}}$ is the amount of time that node i has been charged in this T_c interval, l_i is the amount of charging time allocated to node i in this T_c interval and $l_i - \varphi_{i,t_{curr}}$ is the amount of remaining charging time. Assuming the future energy consumption rate does not change, Equation (2.3) estimates the residual nodal energy at time $(n+1)T_c$.

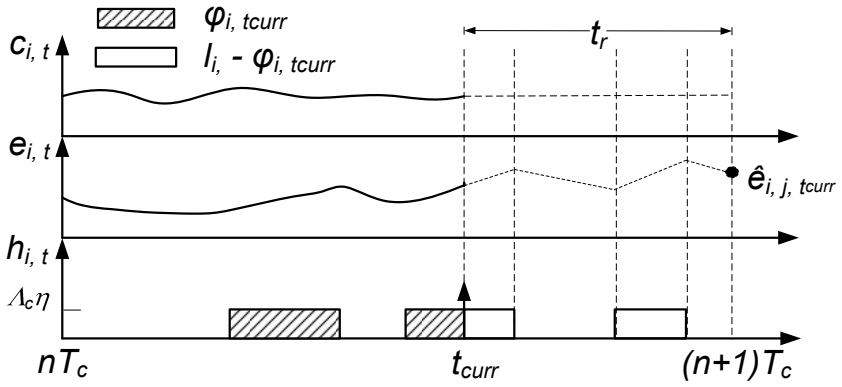


Figure 2.3 Computation of $\hat{e}_{i,j,t}$.

2.3.2 Charging Scheduling Algorithm

Every T_c time, sensor nodes report their nodal status to the BS, including residual energy level ($e_{i,t}$), energy consumption rate ($c_{i,t}$), sensory data packet generation rate ($r_{i,t}$), set of parents on the

energy-minimum paths to the sink ($P_{i,t}$), and the expected number of transmissions to deliver a packet successfully to each parent $p \in P_{i,t}$ ($ETX_{i,p,t}$), based on which the BS schedules the charging activities for the next T_c interval. t is the time when node i reports these information to the BS. The charging scheduling algorithm works in two phases. Firstly, it selects a set of sensor nodes that should be charged in the next T_c interval. Secondly, it determines a sequence in which the sensor nodes are charged so that the movement time is minimized. It also distributes the amount of charging time l_i for the next T_c interval to each node in the sequence.

2.3.2.1 Charging Energy Allocation

To allocate charging energy to sensor nodes, the BS first estimates the future nodal energy consumption rate, denoted as $\hat{c}_{i,t}$, for every sensor node i. Let ρ_i be the percentage of charging energy that should be allocated to sensor node i in one T_c interval. To maximize the network lifetime is to maximize

$$\min_i \left\{ \frac{e_{i,t}}{\hat{c}_{i,t} - \rho_i * \Lambda_c * \eta} \right\}, \tag{2.5}$$

where $0 \leq \rho_i \leq 1$ and $\sum_i \rho_i \leq 1$. Algorithm 1 applies the binary search method to solve the optimization problem.

Algorithm 1 Charging scheduling algorithm to maximize the minimal nodal lifetime

Input: $e_{i,t}$ and $\hat{c}_{i,t}$ for every sensor node i
Output: ρ_i

$low \leftarrow \min \frac{e_{i,t}}{\hat{c}_{i,t}}, up \leftarrow \infty$
/ low/up is the lower/upper bound of the network lifetime */*
$target \leftarrow low$
/ target is the maximum achievable network lifetime */*
while $up - low > \epsilon$ **do**
  calculate ρ_i by solving $target = \frac{e_{i,t}}{\hat{c}_{i,t} - \rho_i * \Lambda_c * \eta}, \forall i \in V$
  if $\sum_{i,\rho_i>0} \rho_i > 1$ **then**
    $up \leftarrow target$
    $target \leftarrow \frac{low+up}{2}$
  else
    $low \leftarrow target$
    $target \leftarrow (up = \infty)?2 * low : \frac{low+up}{2}$
return ρ_i

Next, we discuss how to estimate $\hat{c}_{i,t}$. For a sensor network without energy charging, the energy-balanced routing is favored to extend the network lifetime. The strategy, however, has a side-effect that packets may be routed through less energy efficient paths to the sink when the energy-minimum paths have nodes with low residual energy. Hence, compared to the energy-minimum routing, the energy-balanced routing consumes more energy in transmitting packets. In a sensor network with wireless charging, the MC is able to charge the energy bottleneck nodes. Therefore, energy-minimum paths should be employed more often to improve the energy utilization efficiency in communication and thus to elongate the network lifetime. Based on this observation, the proposed charging scheduling algorithm should intentionally allocate more energy to nodes on energy-minimum paths in order to guide sensor nodes to utilize these paths more frequently. For this purpose, $\hat{c}_{i,t}$ is computed as

$$\hat{c}_{i,t} = \alpha c'_{i,t} + (1 - \alpha) c_{i,t}, \tag{2.6}$$

where $c_{i,t}$ is the actual energy consumption rate reported by node i, $c'_{i,t}$ is the energy consumption rate of node i if all sensor nodes use energy-minimum paths, and α is a value between 0 and 1, called the charging guiding coefficient. In the following, we present how to determine $c'_{i,t}$ and α.

Based on the collected $P_{i,t}$ information from each sensor node, the BS can build a directed acyclic graph. Note that, if sensor node i has multiple energy-minimum paths towards the sink (e.g., several paths from i have the same value of Equation (2.1)), we assume that it transmits packets evenly among these paths. Specifically, if i has k energy-minimum paths, it embeds all k energy-minimum parents and the corresponding link qualities to each parent in the status report to sink. As link qualities are usually stable in a relatively long run [29], we assume that the energy-minimum paths do not change much during one T_c interval as long as T_c value is in a reasonable range. Suppose each sensor node generates a packet at rate $r_{i,t}$ in future, and all sensor nodes use energy-minimum paths to transmit packets. To transmit the packets for itself, the energy consumption rate at sensor node i is

$$\sum_{s \in S_{i,t}} e_{tx} * ETX_{i,n^s_{i,t},t} * \frac{r_{i,t}}{|S_{i,t}|}, \tag{2.7}$$

where $S_{i,t}$ denotes the set of energy-minimum paths from sensor node i to the sink and $n^s_{i,t}$ denotes the

next hop node of i on the energy-minimum path s. To successfully forward packets generated by other nodes, the energy consumption rate at i is

$$\sum_{j \neq i} \sum_{s \in S_{j,t}} (e_{rx} * ETX_{b^s_{i,t},i,t} + e_{tx} * ETX_{i,n^s_{i,t},t}) * \frac{r_{j,t}}{|S_{j,t}|} * I_{i \in s}, \tag{2.8}$$

where $I_{i \in s}$ is an indicator function whose value is 1 if and only if node i is on path s and $b^s_{i,t}$ denotes the previous hop node of i on the energy-minimum path s. e_{tx} and e_{rx} are the expected energy consumed to transmit and receive a packet, respectively, and the values depend on the specific underlying MAC protocols. Hence, $c'_{i,t}$ can be computed by summing up Equation (2.7) and Equation (2.8). Figure 2.4 shows an example of the above procedure. Suppose $\forall i, j, r_{i,t} = 1$ pkt/s, $e_{tx} = e_{rx} = 0.06$ J/pkt and $ETX_{i,j,t} = 1$ except $ETX_{3,sink,t} = 3$ and $ETX_{1,sink,t} = 2$. (a) shows the topology of a network with 10 source nodes and the black square represents the sink. (b) shows the $c'_{i,t}$ value for each node. All routing paths connecting the sensor nodes and the sink are the energy-minimum ones. As node 4 needs two transmissions to reach the sink while node 3 needs three transmissions, path $6 \rightarrow 4 \rightarrow 2$ is the energy-minimum path. Take node 4 for instance; for each of the nodes 7, 8, 9, and 10, it has three energy-minimum paths to the sink and two of them pass through node 4; for node 5, it has two energy-minimum paths to the sink and only one of them passes through node 4; for node 6, it only has one energy-minimum path to the sink and it passes through node 4. Therefore, the energy consumption rate for node 4 to relay the packets for nodes 5, 6, 7, 8, 9 and 10 is $(\frac{2}{3} * 4 + \frac{1}{2} + 1) * (0.06 + 0.06) = 0.5$ J/s, the energy consumption rate for transmitting its own packets is 0.06 J/s; hence we have $c'_{4,t} = 0.56$ J/s.

The value of the charging guiding coefficient α is related to two factors. One factor is the relative charging capability of the MC, which is reflected by the ratio between the amount of energy that can be charged per time unit (i.e., $\Lambda_c * \eta$) and the whole network energy consumption rate (i.e., $\sum_i c_{i,t}$). When the charging capability is relatively strong, e.g., $\frac{\Lambda_c * \eta}{\sum_i c_{i,t}}$ is large, a larger α values is favorable. This means that the charging scheme should guide more packets to be delivered along the energy-minimum paths as the capability of the MC is strong enough to compensate the energy deficiency in time, and accordingly, more energy should be allocated to nodes on the energy-minimum paths. When the capability is low, a smaller α value should be used instead. In addition, the u value also affects the allocation of the chargeable energy. When $u = 1$, each sensor node uses a fixed shortest path to route

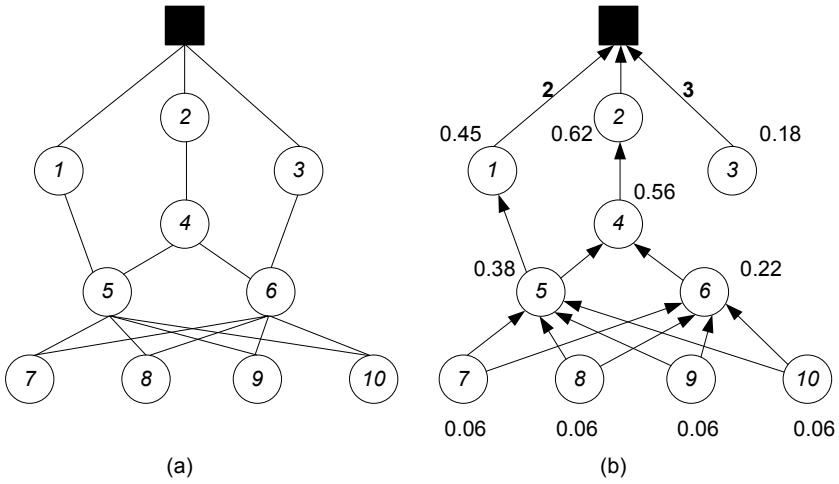


Figure 2.4 An example of calculating $c'_{i,t}$ values.

packets, and therefore is not affected by the charging schedule. When $u > 1$, each sensor node selects its path based on the routing metric. As the routing metric in Equation (2.2) is affected by the charging schedule, the routing decision can be guided by adjusting the charging schedule. Besides, the larger is u, the more effective is the guidance. Considering both factors, we define α as:

$$\alpha = 1 - u^{-\frac{\Lambda_c * \eta}{\sum_i c_{i,t}}}. \tag{2.9}$$

With this formula, when $u = 1$, α is equal to 0 and $c'_{i,t}$ has no impact on the consumption rate estimation. When $u > 1$, the stronger is the relative charging capability, the larger is α and the more weight is given to $c'_{i,t}$ when computing $\hat{c}_{i,t}$.

2.3.2.2 Charging Sequence Determination

In practice, the moving speed of a robot is limited [30] (e.g., between 0.2 and 2 m/s). Too frequent movement may waste time that can be used to charge sensor nodes. Hence, given an allocation plan of charging energy, as computed above, it is important to determine a charging sequence to implement the allocation with as little movement as possible.

The procedure of the charging sequence determination works as follows and an example is given in Figure 2.5. Figure 2.5(a) shows the positions of 5 nodes and triangle 0 stands for the MC. Figure 2.5(b)

gives a naive charging sequence where the MC visits the nodes in the ascending order of nodal lifetime $\frac{e_{i,t}}{\hat{c}_{i,t}}$. The shadow width represents the moving time. Figure 2.5(c) shows the procedure of merging ρ_1 and ρ_4 into ρ_2. Figure 2.5(d) shows the final charging sequence rearranged by the VRPTW solver.

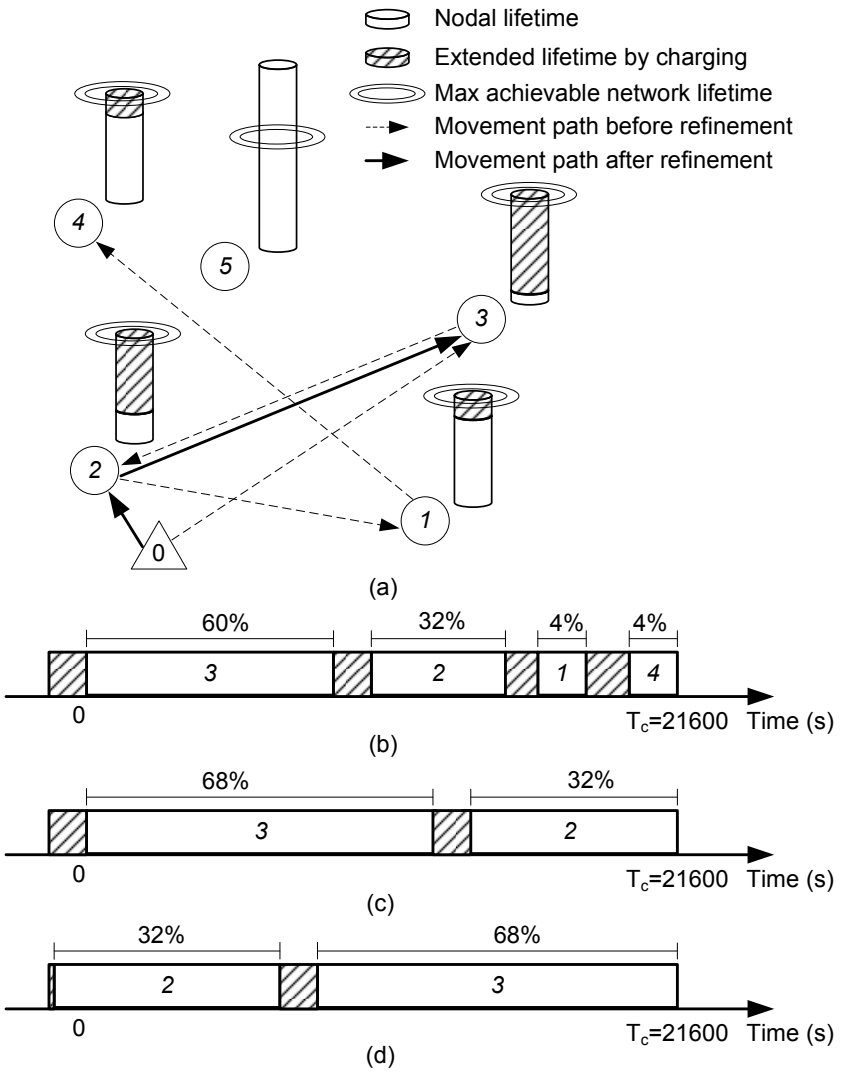


Figure 2.5 An example of the movement refinement.

- Given the percentage of the charging energy ρ_i, the sensor nodes are sorted ascendingly according to their nodal lifetime $\frac{e_{i,t}}{\hat{c}_{i,t}}$. For example, Figure 2.5(a) illustrates the position and nodal lifetime of 5 nodes where the e_i values are 750, 300, 150, 750, 900 J and the $\hat{c}_{i,t}$ values are 0.015, 0.02, 0.03, 0.015, 0.01 J/s. The output of Algorithm 1 produces ρ_i values as 4%, 32%, 60%, 4%, 0% and $target$ as 55714 s assuming $\Lambda_c \eta = 0.045$ J/s. Figure 2.5(b) shows the

sorting result. It also gives us a naive charging sequence with possibly high movement overhead, e.g, $T_e = T_c - T_m^{0,3,2,1,4}$ where $T_m^{0,3,2,1,4}$ is the total moving time along the trajectory $0 \rightarrow 3 \rightarrow 2 \rightarrow 1 \rightarrow 4$ and T_e is the effective charging time. $\rho_i * T_e$ is the amount of charging time allocated to node i.

- The ρ_i value of the maximum lifetime node is iteratively merged to that of the minimum lifetime node until the the battery ceiling of the minimum lifetime node is reached, i.e., $\frac{E_s - e_{i,t}}{\Lambda_c \eta} < \rho_i * T_e$. For example, in Figure 2.5(c), ρ_4 is merged into ρ_3 at first. If the updated ρ_3 does not result in a battery ceiling hit, we update $T_e = T_c - T_m^{0,3,2,1}$ and $\rho_3 = \rho_3 + \rho_4$. Then, the algorithm tends to merge ρ_1 into ρ_3. If the merging leads to a battery ceiling hit, we merge a part of ρ_1 value into ρ_2. This procedure ends when the maximum nodal lifetime is less than T_c or only one node exists after merging.

- VRPTW solver [31], which solves the vehicle routing problem with time window [32], is called to rearrange the visiting sequence to further reduce the movement time. Here, the nodal lifetime is the deadline for each node to be visited. For example, in Figure 2.5(d), the rearranged sequence has $T_m^{0,2,3} < T_m^{0,3,2}$ and $T_e = T_c - T_m^{0,2,3}$. $\rho_i * T_e$ is the amount of charging time allocated to node i and the final charging sequence ready for execution is $\langle\langle 2, \rho_2 T_e\rangle, \langle 3, \rho_3 T_e\rangle\rangle$. Obviously, the amount of effective charging time after the movement refinement is much larger than the one before and thus more energy is replenished into the network.

2.3.3 Performance Upper Bound

Here, we assume the sensory data packet generation rate $r_{i,t}$ of a node does not change during the network lifetime and thus $r_{i,t}$ is denoted as r_i. When the MC's movement delay is ignored and the link qualities are perfect, the optimal solution can be described by the following linear programming formulation.

$$\max \quad T,$$

s.t.:

$$T * r_i + \sum_{j \in N_i} f_{j,i} = \sum_{j \in N_i} f_{i,j}, \tag{2.10}$$

$$T * \sum_{i} r_i = \sum_{j \in N_{BS}} f_{j,BS}, \tag{2.11}$$

$$e_{tx} * \sum_{j \in N_i} f_{i,j} + e_{rx} * \sum_{j \in N_i} f_{j,i} \leq E_s + a_i * \Lambda_c * \eta, \tag{2.12}$$

$$\sum_{i} a_i \leq T, \tag{2.13}$$

$$f_{i,j}, a_i \geq 0. \tag{2.14}$$

Here, T is the network lifetime. $f_{i,j}$ is the total number of packets transmitted from nodes i to j during the network lifetime. a_i is the total amount of time that the MC charges i.

Constraints (2.10) and (2.11) reflect the flow conservation requirements. Constraint (2.12) reflects that the energy used for transmission and reception should be smaller than E_s – the battery capacity of a sensor node – plus the energy charged from the MC. Constraint (2.13) states that the MC could charge one node at a time and thus the total charging time cannot exceed the network lifetime. The output $\langle f_{i,j}, a_i \rangle$ is the joint routing and charging solution. It specifies the number of data packets transmitted over the link (i, j) and the total charging time on node i so that the network lifetime can be maximized.

However, the LP formulation does not take the MC's movement and packet retransmissions into account. Hence, it provides an upper bound of the achievable network lifetime. This formulation is used in both testbed experiment and simulation to evaluate the performance of the proposed J-RoC scheme.

2.4 Design and Implementation

To evaluate the performance of the proposed J-RoC scheme, we have built a prototype system, and the design details of this system are as follows.

2.4.1 Hardware Component

In the prototype system, a Powercast wireless power charger [14] is installed on an Acroname Garcia robot [16] which works as the MC, and a Powercast wireless power receiver is connected to the batteries of a sensor node. The MC communicates with the BS (a PC in the experiments) via an IEEE 802.11b interface to receive the charging scheduling information. When the MC moves into close proximity of a sensor node, the power receiver can collect the energy transferred wirelessly from the MC and use it to charge the batteries of the node.

The energy charging is carried out in the 903-927 MHz band while sensor nodes communicate in the 2.4 GHz band. The power consumption is 3 W when the MC is charging, and the effective amount of energy that can be captured by a receiver varies with the distance between the receiver and the MC; that is, the charging efficiency decreases exponentially when the distance increases. In our system, the MC moves at 1 m/s and the average distance between it and the node charged is about 10 cm which results in 45 mW received power. Note that we use the Powercast products only to evaluate J-RoC's performance in our prototype system.

2.4.2 Software Component

Figure 2.6 shows the software architecture where the shaded parts were elaborated in Section 2.3. The software running on the base station is developed in JAVA, and the sensor node software is developed based on TinyOS 2.1.

In the node software, the *routing engine* module periodically broadcasts beacons containing the information about the energy-minimum path cost and the current routing path cost from the node to the BS. The costs are computed using Equations (2.1) and (2.2) respectively with the latest information of $c_{i,t}, e_{i,t}, \varphi_{i,t}$ and l_i from the *power manager* component and $ETX_{i,j,t}$ from the *link estimator* module. Once receiving a beacon, a node selects the neighbor with the least routing cost to be its next-hop node and updates the energy-minimum parent. The *forwarding engine* module is responsible for forwarding the sensor data packets for the *application* component, and the status reports for the power manager component. The *dissemination engine* module informs the power manager component of the latest l_i value when it receives the charging scheduling messages from the BS. The application component

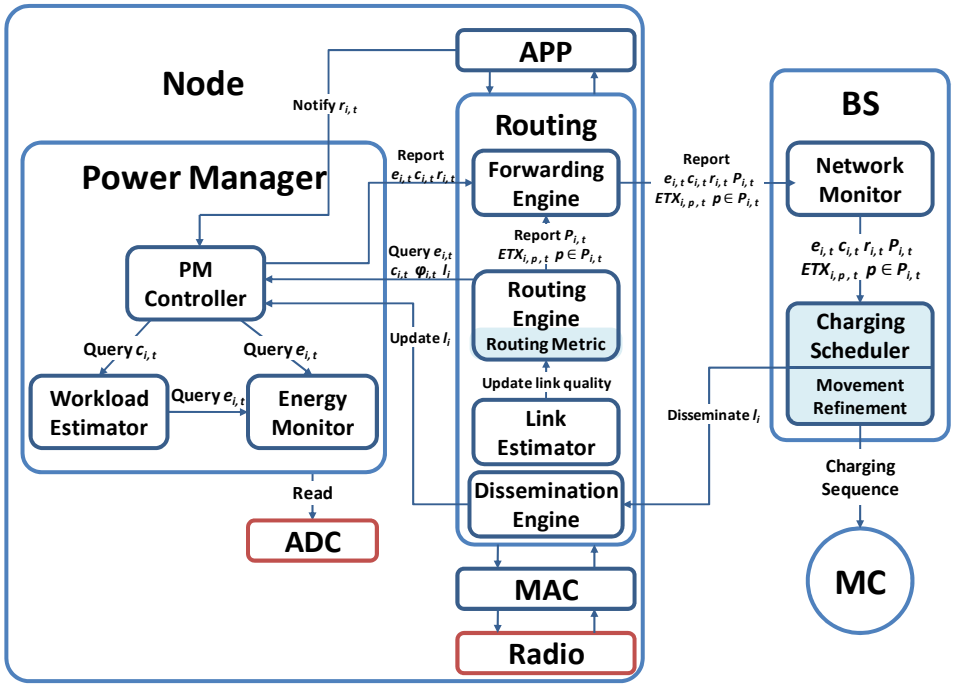


Figure 2.6 Conceptual sketch of the software component.

notifies the power manager component of the future sensory data packet generation rate $r_{i,t}$.

The power manager component boots up automatically with the system and maintains all the charging and nodal energy related information. The *power controller* module records the elapsed charging time $\varphi_{i,t}$ during a T_c interval. It also reports the latest nodal residual energy $e_{i,t}$ (provided by the *energy monitor* module), the energy consumption rate $c_{i,t}$ (provided by the *workload estimator* module) and the $r_{i,t}$ value, together with the $P_{i,t}$ and $ETX_{i,p,t}, p \in P_{i,t}$ (provided by the *routing engine* module) to the BS periodically. In our implementation, the workload estimator module employs the exponentially weighted moving average (EWMA) method to estimate the $c_{i,t}$ values.

At the BS, after the energy reports from each node have been received, the *network monitor* component updates $e_{i,t}$, $c_{i,t}$, $r_{i,t}$, $P_{i,t}$ and $ETX_{i,p,t}, p \in P_{i,t}$ of a node accordingly in a timely manner. Every T_c interval, new charging activities are determined by the *charging scheduler* component with the algorithm described in Section 2.3.2; then, the BS informs the MC of the new schedule and disseminates the messages containing the latest l_i value to the network.

2.5 Experimental Study

2.5.1 Experimental Setup

In the experiments, 10 TelosB sensor nodes are deployed according to the topology shown in Figure 2.4(a). The neighboring nodes are two meters apart and the CC2420 radio transmission power is set to level 3 which results in a 3.5 m communication range. The sink node is connected to a PC with stable power supply and does not need to be charged. During the experiments, each sensor node generates a data packet every 2.5 seconds. A modified X-MAC [33] protocol is run on each sensor node with a Low Power Listening interval of 250 ms and default channel checking time of 50 ms. The T_c length is one hour to reschedule the charging activities.

Each sensor node is powered by two 1.5 V 2000 mAh alkaline rechargeable batteries, and Figure 2.7 shows the mapping between the residual energy and battery voltage level. Particularly, 3000 J energy is consumed in the voltage range 3 V$\sim$2.6 V with running time of 8.4 hours, 7000 J is consumed in the voltage range 2.6 V$\sim$2.2 V in a slower pace with 23 hour running time and 2000 J is consumed in the voltage range 2.2 V$\sim$1.9 V with 5.3 hour running time. The running time is measured with 100% radio duty cycle. The result is achieved through five trials of experiments with 100% radio duty cycle.

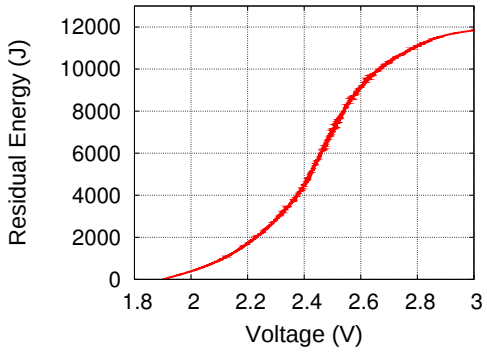


Figure 2.7 Battery energy profile.

To save experiment time, the evaluation is conducted when the voltage varies from 3 V to 2.6 V in which range more serious battery leakage is accompanied as illustrated in Figure 2.7. For each node, the energy level is 100% when the voltage reading is 3 V and the battery is assumed to be completely depleted at voltage level 2.6 V.

2.5.2 Evaluation Results

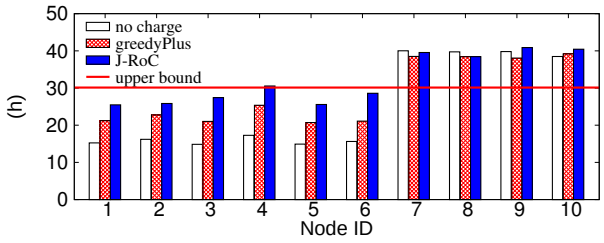
In the experiments, we evaluate (i) the network lifetime upper bound according to 2.3.3 and the actually achieved network and nodal lifetime, when the energy-balanced routing is used without charging (tagged as *no charge* in the figures), the energy-balanced routing combined with greedyPlus scheme [22] is used, and the J-RoC scheme is used, respectively; (ii) the average packet rate (including both the self-generated and the forwarded data packets) of individual nodes; and (iii) the distribution of charging time to individual nodes. As simulation results in [22] have shown that greedyPlus scheme performs better with energy-balanced routing than with energy-minimum routing, we only show the results of greedyPlus with energy-balanced routing in the experiment and simulation evaluations. Parameter u is set to 1000.

2.5.2.1 Overall Evaluation Result of J-RoC

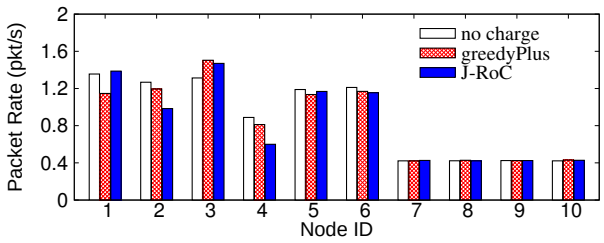
Figure 2.8(a) shows the network lifetime upper bound and the nodal lifetime of individual nodes. The network lifetime upper bound is 30 hours while the achieved network lifetime is 14.9 hours (bounded by node 3), 20.5 hours (bounded by node 5) and 25.5 hours (bounded by node 1) for no charge, greedyPlus and J-RoC respectively. The advantage of J-RoC on prolonging the network lifetime is demonstrated in two aspects in the figure. First of all, compared to the no charge case, the ratio of network lifetime improvement is about 71% (from 14.9 hours to 25.5 hours); compared to the greedyPlus scheme, the ratio of improvement is about 24% (from 20.5 hours to 25.5 hours). Moreover, J-RoC achieves 85% of the network lifetime upper bound (25.5 hours out of 30 hours). Secondly, the J-RoC scheme helps to reduce the standard deviation of the nodal lifetime which results in more efficient usage of the energy. Specifically, the standard deviation of the nodal lifetime is 6.6 hours for J-RoC, 8.6 hours for greedyPlus and 12.3 hours when there is no energy charging.

The improvement in network lifetime shown by Figure 2.8(a) is achieved by guiding nodes to use energy-minimum paths more frequently and allocating more charging energy to nodes on these paths. The average packet rate shown in Figure 2.8(b) and the charging time allocation depicted in Figure 2.8(c) reveal these behaviors in detail.

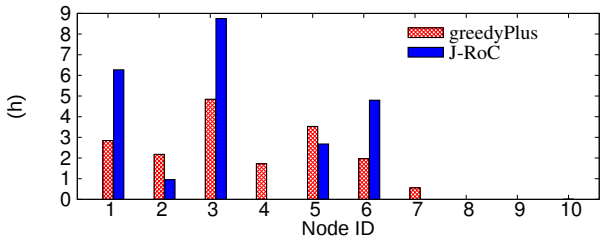
As shown by Figure 2.8(b), nodes 1, 2 and 3 have forwarded quite different numbers of packets



(a) Lifetime of network and individual nodes.



(b) Average data packet rate in individual nodes.



(c) Distribution of charging time among sensor nodes.

Figure 2.8 Experimental results.

when different schemes are used, though they are all one-hop away from the sink. With no charge, these nodes are equally used and their packet rates are all around 1.3 pkt/s because the energy-balanced routing is used. When J-RoC is used, node 2's packet rate drops to 0.95 pkt/s, which is significantly lower than the packet rates of nodes 1 and 3 (i.e., 1.45 pkt/s and 1.55 pkt/s, respectively). When the greedyPlus scheme is used, node 2's packet rate is approaching 1.3 pkt/s and the packet rate of node 3 is much higher than that of node 1 and node 2 respectively. Figure 2.8(c) shows that the charging patterns to nodes 2 and 4 are different when different schemes are used. With greedyPlus, both nodes 2 and 4 are charged with 4 hours in total, but only node 2 is charged in J-RoC and the charging time is less than

1 hour. These differences are attributed to the following reasons. First of all, in greedyPlus, the routing decisions are made without the knowledge of charging activities, and therefore, packets are routed in the energy-balanced manner by using paths through nodes 1, 2 and 3 evenly. The J-RoC scheme, on the other hand, tends to guide nodes to utilize the energy-minimum paths more frequently. Also, if a node has multiple energy-minimum paths that can be used, it is guided to use them in a balanced way. Hence, fewer packets go through node 2, more packets are forwarded by nodes 1 and 3, and the numbers of packets passing nodes 1 and 3 are similar. Secondly, the charging decisions made by greedyPlus is simply to balance nodal lifetimes, without considering routing activities in the network. Therefore, both nodes 2 and 4 are charged with a significant amount of energy as they consume a significant amount of energy to forward packets toward the sink. Differently, the J-RoC scheme makes charging decisions through considering two factors in a balanced manner: guiding nodes to use energy-minimum paths more often, and balancing nodal lifetimes. Consequently, nodes 2 and 4 are seldom charged as they are not on energy-minimum paths and they consume less energy to forward packets than nodes 1 and 3.

In general, the differences in the nodes' packet rates and the allocated charging time among individual nodes reveal the principle behind the design of the J-RoC scheme.

2.5.2.2 Summary

The experimental results have demonstrated the advantage of J-RoC on improving the network lifetime through proactively guiding the routing activities and delivering the energy to where it is needed. When J-RoC is used, more packets are routed through the energy-minimum paths and more charging energy is allocated to nodes on these paths.

2.6 Simulation Study

2.6.1 Simulation Setup

Extensive simulations have been conducted in a custom simulator to evaluate the performance of J-RoC in large-scale networks. In the simulations, 100 nodes are randomly deployed to a 500 m $\times$ 500 m field. The base station and the sink are placed in the center of the field. Table 2.2 lists the default

simulation parameters. As the charging scheduling interval T_c is much larger than the data report interval (default 6 hours compared to 4 minutes), the overhead of the nodal status information collection and charging scheduling information dissemination is neglected in the simulation.

Table 2.2 Simulation Parameters

Parameter	Value
communication range of a sensor node (m)	70
battery capacity of a sensor node: E_s (KJ)	10
energy consumed for MC's charging operation: Λ_c (W)	3
energy consumed for transmitting a packet: e_{tx} (J/pkt)	0.05
energy consumed for receiving a packet: e_{rx} (J/pkt)	0.06
MC's charging efficiency: η (%)	1.5
MC's moving speed: v (m/s)	1
system parameter u	1000
data generation rate: r_i (pkt/h)	15
charging scheduling interval T_c (h)	6

2.6.2 Simulation Results

We measure the network lifetime achieved by the J-RoC scheme, the greedyPlus scheme [22], and the upper bound network lifetime derived in Section 2.3.3 under different scenarios with varying T_c interval, routing metric parameter u, charging efficiency η, data generation rate r_i and the moving speed of the MC v. In order to compare with the upper bound network lifetime whose calculation assumes a fix data packet generation rate over time, we assume $r_{i,t} = r_i$ in the simulation. Note that the calculation of the upper bound of network lifetime does not factor in T_c, u and v; hence, its value remains constant as these parameters change. In addition, we also study the effectiveness of the movement refinement strategy described in Section 2.3.2 through comparing the J-RoC scheme with its naive version that does not have this refinement (tagged as J-RoC-Naive in the figures).

2.6.2.1 Network lifetime with varying T_c

In the proposed scheme, the charging scheduling happens every T_c interval, and the length of T_c affects both the movement overhead of the MC and the amount of energy that an individual node can

be charged. To investigate how the scheduling frequency affects the network lifetime, we first evaluate the performance of all the schemes when the T_c interval changes.

Figure 2.9(a) shows that the network lifetime achieved by J-RoC outperforms greedyPlus and J-RoC-Naive under various T_c values and approaches 95% of the upper bound of network lifetime. In Figure 2.9(a), the lifetimes achieved by both the J-RoC and the greedyPlus schemes decrease slightly as T_c increases. This is due to the fact that the charging decisions are made based on the prediction of the network status for the T_c period and they cannot adapt to the network changes effectively if T_c is long. However, even when T_c is as long as 24 hours in our simulation, J-RoC can still achieve 90% of the upper bound of the network lifetime.

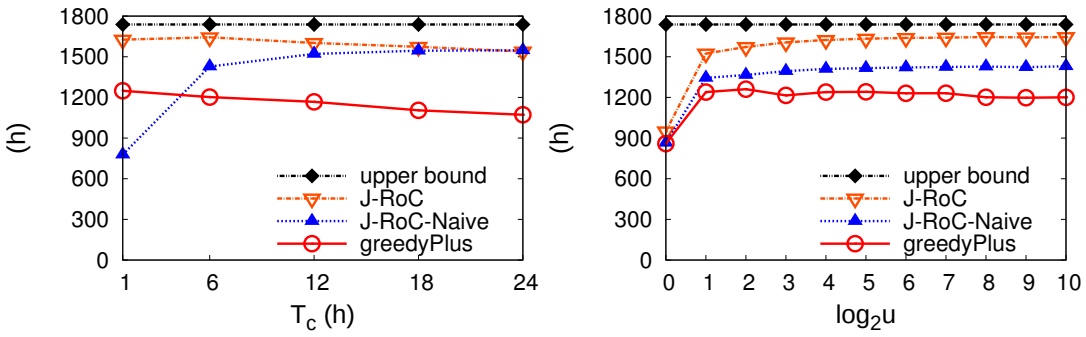
Figure 2.9(a) also shows that the difference between the network lifetime achieved by J-RoC and J-RoC-Naive decreases as T_c increases. This is because the number of nodes to be charged in the J-RoC-Naive scheme is independent of the length of T_c. When T_c increases and charging is scheduled less frequently, the MC stays with a node for a longer time and moves less frequently as well. Therefore, the total movement time decreases and more time could be utilized for charging. Finally, J-RoC and J-RoC-Naive achieve the similar lifetime when T_c is long enough (e.g., 24 hours in the simulation).

2.6.2.2 Network lifetime with varying u

As the value of u affects both the routing metric and the charging schedules in J-RoC, we vary u and measure the achieved network lifetime by all schemes. The results are plotted in Figure 2.9(b).

Compared to $u = 1$, the performance of all schemes improves significantly once u is greater than 1, as the energy-balanced routing avoids depleting the energy of a partial set of nodes and hence elongates the network lifetime. Among them, J-RoC outperforms greedyPlus and J-RoC-Naive under various u values. For instance, when $u = 1024$, greedyPlus, J-RoC-Naive and J-RoC achieve 69%, 82% and 95% of the upper bound of network lifetime, respectively.

When the value of u increases, the performance of J-RoC gradually improves since the charging scheme can guide the routing activities more effectively as described in Equation (2.9). For example, J-RoC achieves 90% of the upper bound of network lifetime when $u = 2$, and achieves 95% of the upper bound when $u \geq 64$.



(a) Network lifetime with varying T_c.

(b) Network lifetime with varying u.

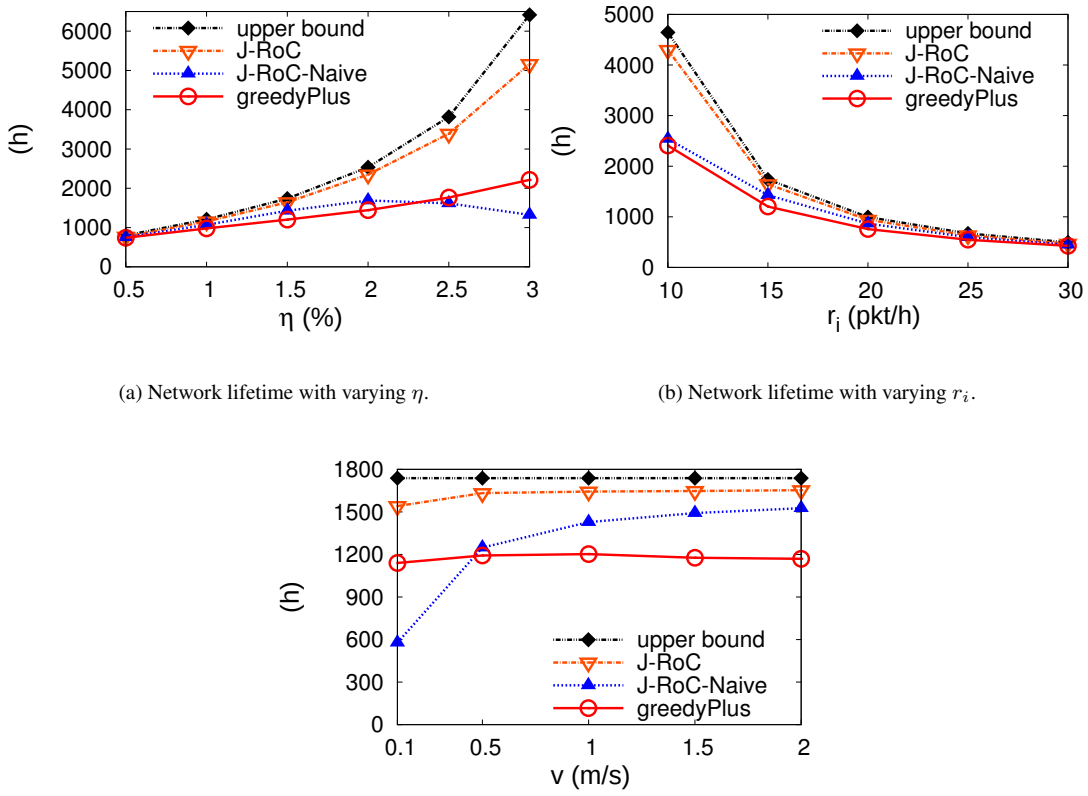
Figure 2.9 Achieved network lifetime comparison with varying T_c and u.

2.6.2.3 Network lifetime with varying η

As the energy charging efficiency (e.g., η) depends on how close the MC could reach each sensor node, we show the performance of all the schemes as the charging efficiency η varies in Figure 2.10(a).

Compared to other schemes, the network lifetime achieved by the greedyPlus scheme ascends the most slowly when η increases. This is because a larger η value allows more energy to be captured by a sensor node. Once a node is charged by the MC, its high nodal energy attracts more traffics, which easily makes itself the energy depletion hot-spot and thus the MC has to keep charging and saving it from being depleted. As the trend continues, the charger is stuck with this node and the opportunities of other nodes to be charged are deprived of. The larger is η, the more intense is this effect. This effect is eliminated in J-RoC which jointly plans the routing and charging activities.

It is also found that the performance of J-RoC-Naive, which does not refine the movement, drops the fastest among all schemes when η is large. This phenomenon can be explained as: the increased η value enables the MC to visit and charge more nodes in one T_c interval, and the movement time increases accordingly without careful movement planning; the J-RoC scheme, on the other hand, alleviates the increasing movement time problem via the movement refinement procedure and outperforms all other schemes.



(a) Network lifetime with varying η.

(b) Network lifetime with varying r_i.

(c) Network lifetime with varying v.

Figure 2.10 Achieved network lifetime comparison with varying η, r_i and v.

2.6.2.4 Network lifetime with varying r_i

Different sensory data generation rates may result in different network-wide distribution of energy and workload, which may affect the performance of J-RoC. To study the impact, we vary the values of r_i, measure the network lifetime achieved by all the schemes and plot the results in Figure 2.10(b).

Compared to other schemes, J-RoC performs the best and well adapts to various distribution of energy and workload. It accomplishes around 94% of the upper bound of network lifetime as the value of r_i varies widely. On the other hand, both greedyPlus and J-RoC-Naive achieve a smaller fraction of the upper bound when r_i is small, e.g., only 58% of the upper bound when $r_i = 10$ pkt/h. This is due to the following reasons. When η is fixed, the smaller is r_i, the stronger is the relative charging capability of the MC. J-RoC can make better use of the relatively stronger charging capability to prolong the

network lifetime, while the performance of greedyPlus may be degraded because the afore-mentioned effect that the MC is stuck to a energy-depletion hot-spot and J-RoC-Naive may waste time and energy for movement.

2.6.2.5 Network lifetime with varying v

In practice, the moving speed of the MC affects the movement time in all the evaluated schemes and the impact is shown in Figure 2.10(c). Obviously, as the moving speed of the MC increases, less time is wasted on the movement and more energy can be replenished into the network. Therefore, the network lifetime achieved by all schemes improves as v increases. Since J-RoC conducts the movement refinement, its performance remains almost the same as v changes and achieves about 95% of the upper bound of network lifetime when $v \leq 0.5$ m/s. On the other hand, J-RoC-Naive approaches 88% of the upper bound when $v = 2$ m/s while the achieved fraction is only 33% when $v = 0.1$ m/s. This result illustrates the effectiveness of the movement refinement in Section 2.3.2.

2.6.2.6 Summary

To summarize, the following observations can be obtained from the simulations:

- Compared to greedyPlus, where the MC only passively makes up for the energy deficiency caused by the routing activities to bottleneck nodes, J-RoC improves the network lifetime more significantly due to its proactive guide on the routing activities. The simulation results also show that J-RoC can effectively approach the upper bound network lifetime under various system configurations.
- The movement refinement strategy helps the J-RoC scheme significantly to reduce the movement overhead and achieve a longer network lifetime compared to J-RoC-Naive and greedyPlus.

2.7 Conclusions

In this section, we propose a practical and efficient joint routing and charging scheme, called J-RoC, to prolong the sensor network lifetime. We present the design and implementation of the J-RoC scheme and evaluate its effectiveness and advantage on prolonging the sensor network lifetime

through both experiments on a prototype system and simulations in large-scale networks, under various configurations. The results show that, through proactively guiding the routing activities and delivering energy to the most energy-demanding places in a joint way, the J-RoC scheme can extend the sensor network lifetime significantly.

Some more issues are left open for future research. For example, the geographical conditions may constrain the movement trajectory of the MC and make some nodes inaccessible. This issue will be factored into the J-RoC scheme. In addition, the J-RoC scheme is designed for a single charger. How to schedule multiple chargers simultaneously is an interesting and more complicated problem, which will also be studied in the future. To evaluate the performance of J-RoC more thoroughly, more theoretical analysis and experiments on larger scale sensor networks will be conducted as well.

CHAPTER 3. JOINT CHARGING AND RATE ALLOCATION FOR UTILITY MAXIMIZATION IN RECHARGEABLE SENSOR NETWORKS

3.1 Introduction

3.1.1 Motivations

Maximizing the network utility, which is a non-decreasing function of nodal sensory data generation rate [17, 18], is critically important when the networks are deployed for monitoring applications. For such networks, the major constraint for utility maximization is the limited energy supply of sensor nodes. One solution to this problem is to devise ambient energy harvesting techniques such as solar [8], wind [10], and vibration [11] to replenish energy into the network at run-time. Although ambient energy is free and green, harvesting-based approaches [9, 34–36] are forced to adapt nodal sensory data rates to the ambient energy availability, which it usually temporal-spatially varying and uncontrollable. Hence, the resulted network utility may not be optimized. For example, a close-to-sink node with low energy harvesting rate may fail to relay data packets generated in its subtree.

Complementary to ambient energy harvesting, the emerging wireless charging technology [14] creates a perpetual power source to provide power-over-distance, one-to-many charging, and controllable wireless power. Applications of this technology to sensor networks have been reported in [37, 38]. In these applications, a mobile charger (MC), which is a mobile robot carrying a wireless charger, moves around the deployment field and transfers energy to sensor nodes wirelessly. With the controllable power source, energy can be delivered to where it is needed. Therefore, sensory data rate can be adjusted proactively to achieve the optimal network utility. However, the energy transfer range is usually limited and the effective amount of energy that can be captured by a node decreases exponentially as the distance between the node and the MC increases [15, 22]. In many practical deployment scenarios,

the MC can only move along some pre-determined trajectories and hence nodes deployed far away from the trajectories may not be charged.

As the aforementioned ambient energy harvesting (free and universally available) and wireless charging (controllable but only a subset of nodes can be charged) technologies are complimentary to each other, it is naturally an attractive idea to combine them for network utility maximization. Ideally, to maximize the network utility, the sensory data rates and charging activities shall be jointly scheduled while taking the ambient energy availability into account. On one hand, the wireless charging energy shall be allocated in the manner such that nodes who contribute more for the network utility increase shall have enough energy to continue such contribution. On the other hand, the sensory data rates shall be determined carefully so that each node can efficiently utilize the replenished energy while not overusing it and thus degrading the network operational time. When scheduling the charging activities and sensory data rates, the temporal-spatially varying ambient energy availability and the fact that the MC can only move along pre-determined trajectories shall not be neglected.

To address this challenging problem, it is important to develop an innovative approach to jointly optimize the tightly coupled three components: (i) charging activities, (ii) sensory data rate, and (iii) ambient energy availability. This is also the focus of the study. Particularly, given a harvesting-enabled and wireless-chargeable sensor network with a pre-determined trajectory, our goal is to maximize the network utility while guaranteeing the network sustainability, which means that sensor nodes shall always maintain positive energy levels to avoid operation interruption.

3.1.2 Literature Survey

3.1.2.1 Rate allocation in ambient energy harvesting sensor networks

Various ambient energy sources like solar [8], wind [10], thermal [39] and vibration [11] have been employed to build sustainable sensor networks. Among them, solar is the most thoroughly studied one.

To deal with the time-varying nature of the solar energy, many rate allocation schemes have been explored [9,34,40,41] to adapt sensory data rates to solar energy availability. In [9], authors extend the framework in [42] by including solar energy and propose to maximize the lexicographic rate of solar rechargeable sensor nodes. The rate assignment problem is formulated as a linear optimization prob-

lem and solved optimally by both centralized and distributed algorithms. [34] proposes a rate control approach for a single energy harvesting node to achieve a series of objectives including the maximization of average sensing rate over time. These objectives are formulated as optimization problems and solved with multi-parametric control algorithms. [40, 41] propose a flow control algorithm for energy harvesting sensor networks. [41] proposes an energy budgeting algorithm that defines the amount of energy a node can use for each epoch. The derived energy budget assignment is optimal in the sense that the variance of energy assigned across epochs is minimized. Using this formulation, the flow control algorithm in [40] maximizes the amount of data collected from the network, given that no node consumes more energy than the assigned budget. However, these works assume that the system utility increases linearly with the sensory data rates. However, for many applications, the utility increases in a sub-linear fashion as the data rates increase (the diminishing return principle).

Using this observation, [35, 36] model the system utility as a concave and non-decreasing function of data rates. Particularly, [35] proposes a dual decomposition and sub-gradient based algorithm, called QuickFix, to compute the data sampling rates. To deal with the issue that the convergence rate is slower than the energy fluctuation rate and thus the possible battery outage and overflow scenarios, a local algorithm, called SnapIt, is designed to adapt the sampling rates with the objective of maintaining the battery at a target level. [36] proposes both a centralized and a distributed scheme to maximize the total utility achieved by all the nodes while ensuring eternal network lifetime. Different from JCRA, all these works are forced to passively adapt sensory data rates to the ambient energy availability, which it uncontrollable. Hence, the resulted network utility may not be optimized,

3.1.2.2 Wireless charging applications

One potential problem with ambient energy harvesting based protocols is that they need to passively adapt to the availability of the ambient energy. In order to actively control the network behaviors, a mobile robot carrying a wireless charger (MC) [14, 15] has been employed to deliver energy to where it is needed. Based on the wireless charging model, existing works can be classified into two categories.

Particularly, Peng et al. [22] build a prototype system to study the feasibility of using the wireless charging technology to prolong the sensor network lifetime. A greedy algorithm is develop to dispatch

the MC to charge a selected set of lifetime bottleneck sensor nodes. Based on the assumptions that a single MC's capability is strong enough to maintain eternal network lifetime, the traffic pattern is fixed and the communication channels are perfect, Shi et al. [25] conduct a theoretical study and propose a static, centralized scheme to maximize the ratio of the MC's vacation time over each renewable energy cycle. To handle practical issues like limited charging capability, dynamic and imperfect communication environment, and heterogeneous node attributes, a joint routing and charging scheme, named J-RoC is proposed in [37] to maximize the network lifetime. Compared with the charging model used in JCRA, [22, 25, 37] assume that the MC can only charge one node at a time, which does not fully utilize the broadcast nature of the wireless energy transfer.

Based on the simultaneous charing model, authors in [43] propose to jointly determine the sensor node deployment and routing strategies to minimize the overall energy consumption at the chargers. The problem is formulated and proven to be NP-complete. Partitioning the two-dimensional deployment filed into adjacent hexagonal cells and requiring the MC to charge sensor nodes from the center of a cell, [44] jointly optimizes the traveling path, routing flow and charging time. A provably near-optimal solution for any desired level of accuracy is developed. [23] studies the energy provision problem and discusses how to deploy the WISP readers so that the WISP tags can harvest sufficient energy for continuous operation. In the same framework, [45] explores how to minimize the time required to charge all RFID tags to a target energy level with a given RFID reader while [46] studies the on-demand charging problem. Although schemes in [23, 45, 46] are designed for RFID systems, similar ideas can be applied to sensor networks as well. However, all the aforementioned works assume that the MC can move freely in the deployment field. Such assumption may not hold in practice.

[38] studies the problem of co-locating the mobile base station on the MC and the MC can only move along some pre-determined trajectories. However, its goal is to minimize the energy consumption of the entire system while ensuring none of sensor nodes runs out of energy, instead of maximizing the network utility. Moreover, as ambient energy harvesting is not considered in [38], to achieve eternal network lifetime, all nodes have to be deployed near the trajectories. Therefore, application of [38] is limited.

3.1.3 Contributions

In this study, we propose a scheme, called JCRA (*J*oint *C*harging and *R*ate *A*llocation), to jointly optimize the charging activities and data rates allocation to maximize the network utility while guaranteeing perpetual network operation. Particularly, through periodical information exchanging between sensor nodes and the MC in JCRA, the MC keeps track of the global energy status of the network, schedules charging activities accordingly, and disseminates the charging schedule to the network. Meanwhile, sensor nodes in a neighborhood adjust their data rates in a collaborative manner, with the target of improve the total neighborhood utility. As such coordination happens in all neighborhoods, the network utility may be improved. The contributions of this work are summarized as follows.

- To the best of our knowledge, JCRA is the first design to maximize the network utility in a MC movement confined, ambient energy harvesting and wireless charging enabled sensor network with the network sustainability requirement.
- We formulate the joint charging and rate allocation problem and develop a centralized, off-line solution, which provides a theoretical foundation to the proposed JCRA scheme.
- JCRA has been thoroughly evaluated via ns-2 simulations. It approaches the performance of the centralized, off-line solution under various network settings while guaranteeing the sustainability requirement.

3.2 System Model

As illustrated in Figure 3.1, our system is composed of three main components: a mobile charger (MC) which can only move along a pre-paved trajectory, a network of sensor nodes equipped with both wireless power receivers and ambient energy harvesting devices like solar panels, and a base station (BS) that collects data from and monitors the energy status of the network as well as directs the MC to charge sensor nodes.

The system works as follows. Each sensor node determines its sensory data generation rate and sends the periodically generated data hop-by-hop to the BS. Besides harvesting ambient energy, a node can receive wireless charging energy from the MC if their distance is less than a certain threshold.

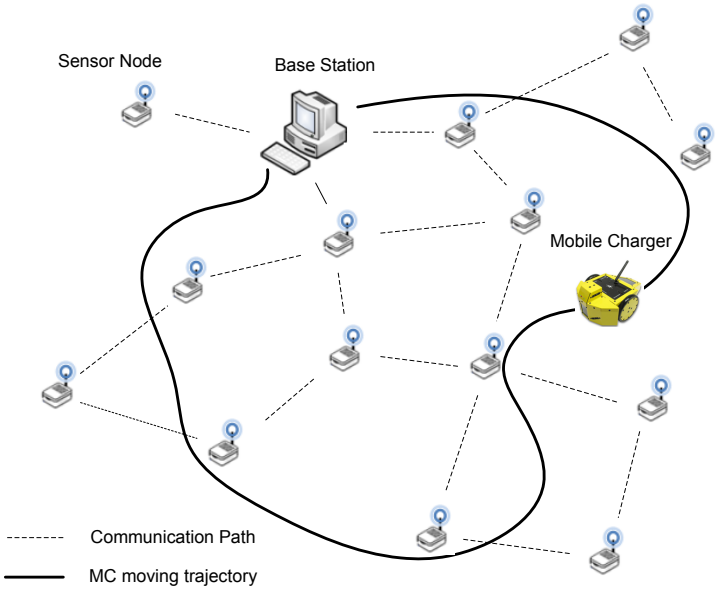


Figure 3.1 System Overview.

Wireless charging model details are described in Section 3.2.2. MC charging activities are scheduled by the BS. To facilitate the scheduling, each node measures its local energy level, estimates its future ambient energy harvesting rate based on a historical ambient energy profile, and reports these information to the BS. With the collected information, the BS schedules the charging activities and commands the MC via a long range radio to execute the schedule. The MC then travels along the trajectory to charge sensor nodes. Meanwhile, the BS also disseminates the schedule to sensor nodes to help them determine their data generation rates. Charging activities are scheduled and executed round by round. In each round, the MC starts from the BS with a full energy battery, moves along the trajectory, charges sensor nodes nearby, and returns to the BS at the end to replace the battery before energy depletion. The system objective is to maximize the network utility (defined in Section 3.2.1) while satisfying the sustainability requirement (defined in Section 3.2.3) .

3.2.1 Network Utility Model

Network utility in many sensor network applications [17, 18, 36] presents the diminishing return property, which means that the network utility increases in a sub-linear fashion as the sensory data rate increases. For instance, in an intrusion detection system, as humans can only move at a certain

speed, increasing the sensing rate above a specific threshold only marginally increases the utility of the application. For this reason, the network utility model used in the work is:

$$\sum_i U(r_i), \tag{3.1}$$

where r_i is node i's sensory data generation rate. $U_i(\cdot)$ is an application specified non-decreasing, concave utility function, e.g., $\forall \Delta > 0, U(x+\Delta) - U(x) > U(y+\Delta) - U(y)$ if and only if $x < y$.

3.2.2 Wireless Charging Model

We assume that the MC can only charge nodes when it stops somewhere along the trajectory. Denote $\Lambda_i(s)$ the wireless charging power received by node i when the MC is at point s on the trajectory, we have

$$\Lambda_i(s) = \begin{cases} \eta_i(s)\Lambda_c & : d(i,s) \leq D_{max}, \\ 0 & : \text{otherwise}, \end{cases} \tag{3.2}$$

where $\eta_i(s)$ is the wireless energy transfer efficiency, Λ_c is the charging power output at the MC and D_{max} is charging distance threshold. When the distance between node i and the MC, namely $d(i,s)$, is larger than D_{max}, the received charging power becomes zero. In general, $\eta_i(s)$ is a decreasing function of $d(i,s)$. In this work, efficiency model in [15] is employed.

3.2.3 Network Sustainability

A sustainable network [47] requires perpetual network operation and no node runs out of energy forever. As the charging activities in the system are scheduled round by round, to maintain network sustainability, we require that (i) no node runs out of energy during each charging round and (ii) the nodal residual energy at the end of a charging round is no smaller than E_t. E_t is a system parameter. The amount of E_t energy serves as the energy buffer to tolerate environmental and network uncertainties, like weather changes, MC cannot move to the exact charging point as expected and packet retransmissions due to deteriorated channel conditions.

3.3 Analytical Study

To provide a theoretical foundation for the proposed JCRA scheme, in this section, we first formulate the joint charging and rate allocation problem and then develop a centralized, off-line algorithm with provably $(1-\epsilon)$ approximate ratio to solve the formulated problem. ϵ is the system tolerable inaccuracy level. The analytical study results will be used in JCRA to schedule the charging activities (see Section 3.4.1 for details). Notations commonly used in this study are summarized in Table 3.1.

Table 3.1 Notation Summary

notation	meaning
$e_i(s_k)$	node i's residual energy when MC is at segment s_k
$g_i(s_k)$	number of packets that i generates when MC is at s_k
$c_i(s_k)$	amount of energy that i consumes when MC is at s_k
$\Lambda_i(s_k)$	i's charging rate when MC is at s_k
$H_i(t)$	i's ambient energy harvesting rate at time t
$\hat{H}_i$	i's estimated average ambient energy harvesting rate till the end of the current charging round
e_i	i's current residual energy level
r_i	i's current data generation rate
$a(s_k)$	amount of time that the MC stops and conducts charging at s_k
τ	time length of a charging round
E_t	required minimal nodal energy at the end of a charging round
E_{max}	nodal battery capacity

3.3.1 Problem Formulation

As the MC moving trajectory S is continuous, in the formulation, we first discretize it into K segments with equal length and use the central point of a segment to represent it. Then, the joint charging and rate allocation problem over one charging round can be formulated as follows:

$$\max \sum_i U\left(\frac{\sum_{k=1}^{K} g_i(s_k)}{\tau}\right),$$

s.t.

$$\tau = \sum_{k=1}^{K} \tau(s_k), \tag{3.3}$$

$$\tau(s_k) = \frac{D(s_k)}{V} + a(s_k), \tag{3.4}$$

$$c_i(s_k) = (e_{tx} + e_{rx}) \sum_{j \in D_i} g_j(s_k) + (e_{tx} + e_{sx}) g_i(s_k) \tag{3.5}$$

$$e_i(s_{k+1}) \leq e_i(s_k) - c_i(k) + H_i(s_k)\tau(s_k) + a(s_k)\Lambda_i(s_k) \tag{3.6}$$

$$e_i(s_{K+1}) \geq E_t, \tag{3.7}$$

$$a(s_k),\ g_i(s_k),\ 0 \leq e_i(s_k) \leq E_{max}. \tag{3.8}$$

The objective is to maximize the network utility in a charging round. $g_i(s_k)$ is the amount of data packets that node i generates when the MC is at segment s_k and τ is the time length of a charging round. τ is a system parameter. As stated in Equation (3.3), τ is the sum of $\tau(s_k)$, the time that the MC spends over each s_k. $\tau(s_k)$ includes two parts, the moving part $\frac{D(s_k)}{V}$ and the charging part $a(s_k)$. $D(s_k)$ is the length of s_k and V is the MC's moving speed. Denote e_{tx}, e_{rx}, e_{sx} the energy cost to transmit, receive and generate a data packet, respectively, Equation (3.5) computes the amount of energy that node i consumes when the MC is at s_k. D_i is node i's descendant node set on the collection tree. $e_i(s_{k+1})$ in Inequality (3.6) is i's residual energy level when the MC enters s_{k+1}. It shall be no more than $e_i(k)$ – i's nodal energy when the MC enters s_k, minuses the amount of energy consume and pluses the energy received at segment s_k. Note that $e_i(s_{k+1})$ could be less than the right hand side of Inequality (3.6) due to energy overflow[1]. Here, $H_i(s_k)$ is i's average ambient energy harvesting rate when the MC is at s_k and $\Lambda_i(s_k)$ is i's charging rate when the MC is at s_k. Finally, Equation (3.7) states the sustainability requirement. In the formulation, both s_0 and s_{K+1} represents the BS. The output $\{a(s_k), g_i(s_k), 1 \leq k \leq K\}$ is the solution for the joint charging and rate allocation problem. It specifies both the charging behaviors $a(s_k)$ and data rate allocation strategy $g_i(s_k)$.

[1]Once a battery is charged to its capacity, its energy cannot be further increased.

In the above formulation, we assume that each node has a historical ambient energy profile, based on which, the average ambient energy harvesting rate when the MC is at s_k (denoted as $H_i(s_k)$ in Inequality (3.6)) can be calculated. For stable energy sources, $H_i(s_k)$ can be treated as a constant value when k varies. For time-varying energy sources like solar, wind, etc., the value of $H_i(s_k)$ depends on when the MC arrives at s_k and how long it stays there. To ease the analysis, we use $\hat{H}_i$, the average ambient energy harvesting rate till the end of the current charging round, to represent $H_i(s_k), 1 \leq k \leq K$. In Section 3.4.1, we will describe how to deal with the time-varying energy sources in practice. Therefore, we can re-write Inequality (3.6) as

$$e_i(s_{k+1}) \leq e_i(s_k) - c_i(k) + \hat{H}_i \tau(s_k) + a(s_k)\Lambda_i(s_k). \tag{3.9}$$

Based on the facts that (i) all constraints in the formulation are linear, (ii) the utility function $U(\cdot)$ is a non-decreasing concave function and (iii) the sum of concaves functions is also concave, it is easy to derive that the above formulation is a convex optimization problem. Solvers likes [48] can be used to solve it.

3.3.2 Approximate Algorithm

The optimization strategy developed in [38] is adopted to construct the approximate algorithm proposed in this section. Particularly, in the above discretized formulation, the number of segments K affects the accuracy of the solution. To determine an appropriate K value, in this section, a lower bound and an upper bound formulations are constructed with a given K value. As shown later, when the K value increases, the network utility achieved by the lower and upper bound formulations will increase and decrease, respectively. Hence, as we keep increasing the K value until the gap between the lower and upper bound network utility is less than a small threshold ϵ, the final solution of the lower bound (or the upper bound) formulation is a $(1 - \epsilon)$ approximation to the original problem.

3.3.2.1 Lower and Upper Bound Formulations

Each segment s_k contains many points. The wireless energy charging rate $\Lambda_i(p), p \in s_k$ varies at different points in the same segment. The idea of building the lower bound formulation is to use the

lowest charging rate on s_k to present $\Lambda_i(s_k)$. Specifically, in the lower bound formulation (denoted as P_{lb}), we set $\Lambda_i(s_k)$ in Equation (3.9) as

$$\Lambda_i(s_k) = \Lambda_i^l(s_k) = \min_{p \in s_k}\{\Lambda_i(p)\}, \tag{3.10}$$

and make all rest equations unchanged. Apparently, solutions of P_{lb} is a lower bound of the discretized formulation. Similarly, we set

$$\Lambda_i(s_k) = \Lambda_i^u(s_k) = \max_{p \in s_k}\{\Lambda_i(p)\}, \tag{3.11}$$

to construct the upper bound formulation (denoted as P_{ub}). As both Equations (3.10) and (3.11) are linear, P_{lb} and P_{ub} are still convex optimization problems and they can be solved by solver [48].

3.3.2.2 Approximate Algorithm

Details of the approximate algorithm is illustrated in Algorithm 2. $\chi = \{\phi, a(s_k), g_i(s_k)\}$ denotes the solution of the problem formulation where ϕ represents the achieved objective value.

Algorithm 2 Approximate Algorithm for Joint Charging and Rate Allocation Problem

Input: $\{e_i(s_1), \hat{H}_i\}$
Output: χ^*

```
K ← K_init
repeat
    Evenly divide trajectory S into K segments s_1, ..., s_K
    χ^l ← solve P_lb
    χ^u ← solve P_ub
    K ← 2 × K
until φ^u / φ^l < 1 + ε
return χ^l
```

In Algorithm 2, it checks the gap between the lower and upper bound objective values. If the gap between them is larger than a system specified accuracy ϵ, the algorithm will double the segment number K. As proved in Lemma 3.3.1, when K increases, ϕ^l will increase and ϕ^u will decrease. Hence, the gap between them will decrease. Iteration continues until the gap in less than ϵ. Obviously, the final output is a $(1-\epsilon)$ approximate of the original joint charging and rate allocation problem

without discretization.

Lemma 3.3.1 *As segment number K increases, ϕ^l increases and ϕ^u decreases.*

proof: 1 *The proof can be done by construction. When we increase K to $2K$, we split each segment $s_k, 1 \leq k \leq K$ into two equal length smaller segments s_k^1 and s_k^2. For P_{lb}, we construct a feasible solution $\hat{\chi}^l(2K)$ on the new segment set and prove that $\hat{\chi}^l(2K) > \chi^l(K)$. As $\hat{\chi}^l(2K)$ is only a feasible solution, we have $\chi^l(2K) > \hat{\chi}^l(2K) > \chi^l(K)$ and the proof is done.*

One observation is that when we split the segment into two small ones, either $\Lambda_i^l(s_k^1) > \Lambda_i^l(s_k)$ or $\Lambda_i^l(s_k^2) > \Lambda_i^l(s_k)$, depending on where the lowest charging rate point appears. Without loss of generality, assume that the lowest charging rate point is in s_k^2 and thus we have $\Lambda_i^l(s_k^1) > \Lambda_i^l(s_k)$ and $\Lambda_i^l(s_k^2) = \Lambda_i^l(s_k)$. Obviously, each node will receive more charging energy when we increase K to $2K$. More charging energy means that sensor nodes can generate and relay more data packets to the BS. Hence, the network utility ϕ^l will increase. Similarly, we can prove that when we split each segment into two smaller segments, ϕ^u value decreases.

3.4 The proposed JCRA scheme

Algorithm 2 is a centralized, off-line algorithm. Directly disseminating data rates to the network requires a unicast from the BS to each sensor node, which is not acceptable for resource constrained sensor networks. Moreover, as network conditions like channel quality and ambient energy harvesting rate are highly dynamic and hard to predict in nature, a distributed, on-line scheme is needed. For this purpose, we propose the protocol called JCRA in this section.

In JCRA, the BS schedules the charing activities and each node determines its own data generate rate in a distributed manner. Figure 3.2 overviews how the data rate is determined. To ease the presentation, we use the topology shown in Figure 3.3 as an example to explain the design details of JCRA.

In general, when node i receives a data packet from its child node or an ACK from its parent node, it extracts the control information embedded in the packet and feeds them into the *Rate Allocator*. Here, the control information includes two items from each child node j of i: j's estimated energy at the

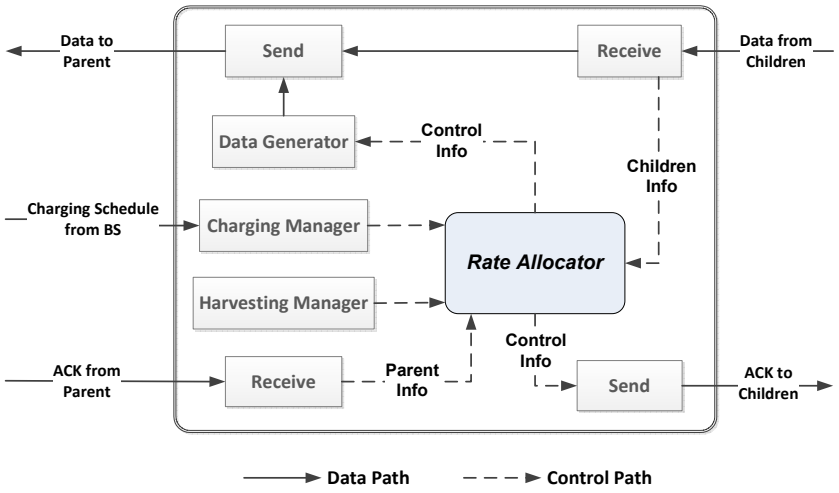


Figure 3.2 Data rate determination overview in JCRA.

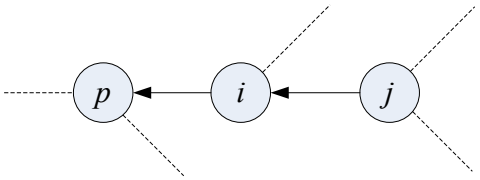


Figure 3.3 Topology used to describe JCRA details.

end of the current charging round $\hat{e}_j$ and its current data generation rate r_j; and one item from parent node p of i: $\Delta_{p,i}$. We will explain the meaning of $\Delta_{p,i}$ in Section 3.4.2.2. The *Harvesting Manager* is responsible to estimate the future ambient energy harvesting rate $\hat{H}_i$. Based on the historical ambient energy profile, schemes like [49] can be employed for $\hat{H}_i$ estimation. The *Charging Manager* updates the charging schedule when node i receives a new schedule from the BS, and removes the parts that have already been executed by the MC. It provides Rate Allocator the rest time length of the current charging round ω and the amount of charging energy that node i can receive during ω time, denoted as $\hat{A}_i$.

With these information, Rate Allocator decides how node i and its child nodes shall adjust their data generation rates to improve the total utility in the neighborhood. The decision is then piggybacked into the ACK packet to its child nodes. Upon being notified, each child node adjusts its own data rate.

3.4.1 Charging Scheduling

Charging activities in JCRA are scheduled by Algorithm 2. To provide necessary information for such scheduling, every sensor node reports e_i and $\hat{H}_i$ values to the BS. Such information can be piggybacked to the generated sensory data packets to save bandwidth and energy cost. As the harvesting rate of some energy sources, e.g., solar, is time-varying, the BS re-runs Algorithm 2 every certain interval I_{MC} with the most updated e_i and $\hat{H}_i$ values to adapt charging activities. I_{MC} is a system parameter. When BS re-runs the algorithm, charging round time length τ and the trajectory S are replaced the rest time length of the charing round ω and the rest of trajectory that the MC has not been visited. In the output of Algorithm 2, the charging schedule $\{a_{s_k}\}$ is then disseminated to all nodes in the network to help them determine their data generation rates. Unlike data rate information which is node specific, the charging schedule dissemination can be achieved via flooding, which is a much cheaper operation than unicasting to each node, especially when the network size is large.

3.4.2 Data Rate Allocation

Data rate allocation in JCRA operates in two phases: *initial phase* and *adaptation phase*.

3.4.2.1 Initial Phase

After the data collection tree has been established, each node needs to set its initial r_i value. Considering the diminishing return property, to achieve good initial network utility, the objective is to evenly allocate the data rates without violating the sustainability requirement. In the following, we use an example (as shown in Figure 3.4) to explain how the initial phase works. In this example, $E_t = 100J$, $e_{tx} = 0.05J/pkt$, $e_{rx} = e_{sx} = 0.01J/pkt$, $\tau = 4h$.

- Each node reports its e_i and $\hat{H}_i$ values to the BS, and derives the number of descendant nodes $|D_i|$ by counting the number of reports it has relayed as shown in Figure 3.4(a).
- With the collected e_i and $\hat{H}_i$ values, the BS runs Algorithm 2 and disseminates the first round charging schedule $\{a(s_k)\}$ to all nodes. In Figure 3.4(b), the trajectory is split into four segments.

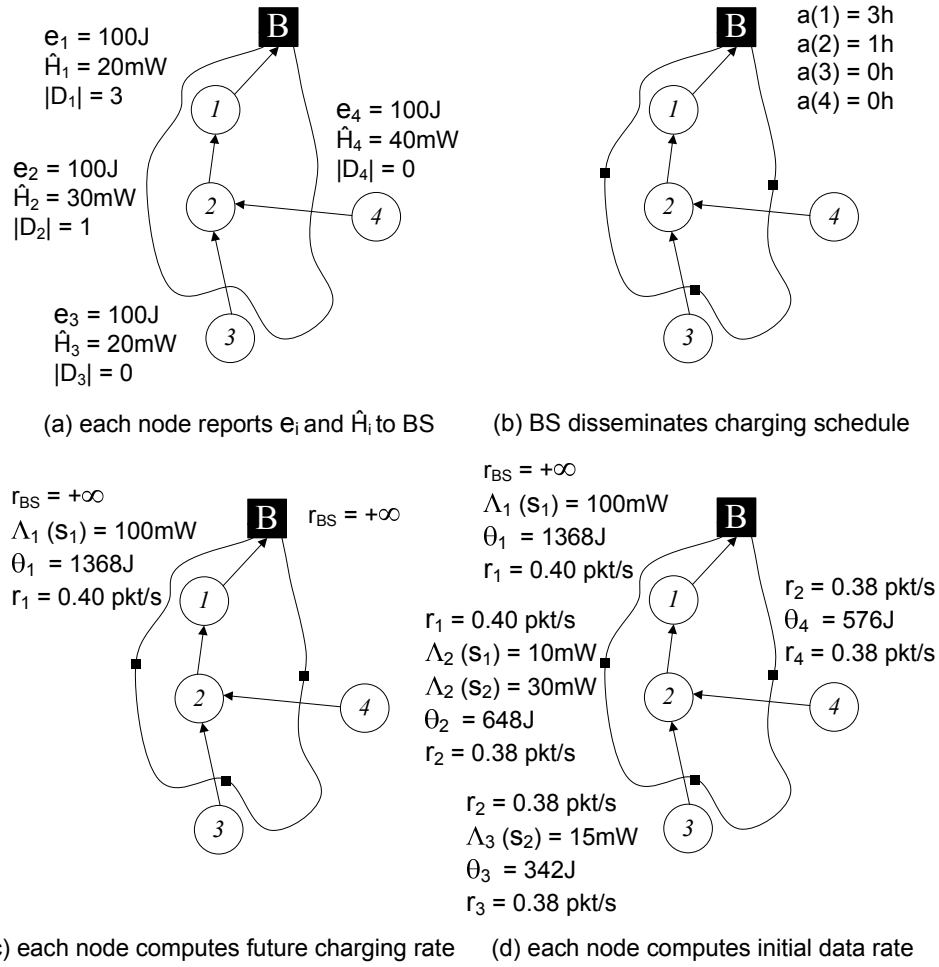


Figure 3.4 An example of JCRA initial phase.

- Once received the charging schedule, a node computes the charging rate $\Lambda_i(s_k)$ for each segment s_k. To compute $\Lambda_i(s_k)$, the geometric center of s_k is used. With $\Lambda_i(s_k)$ information, a node estimates the total amount of energy it can receive in the current charging round as

$$\theta_i = \hat{H}_i \tau + \sum_{k=1}^{K} a_(s_k)\Lambda_i(s_k). \tag{3.12}$$

In Figure 3.4(c), the charging rate at node 1 is 100 mW when the MC charges at the first segment. When MC charges at other segments, the charging rate is zero as the charging distance is beyond the threshold D_{max}. Hence, $\theta_1 = 1368$ J in the next four hours. To evenly allocate the data rate without violating the sustainability requirement, a node assumes that itself and its descendant

nodes have the same initial data generation rate as

$$\frac{e_i - E_t + \theta_i}{((e_{tx} + e_{sx})|D_i| + (e_{tx} + e_{sx}))\tau}. \tag{3.13}$$

In order to guarantee that its parent node is not overloaded, initial data rate is determined as

$$r_i = \max\{0, \min\{r_p, \text{Equation (3.13)}\}\}. \tag{3.14}$$

r_p is the initial data rate that p determines for itself and its descendant nodes. In Figure 3.4(c), as the BS has infinite energy supply, it notifies its children nodes with $r_{BS} = +\infty$. Node 1 computes $r_1 = 0.40$ pkt/s and sends r_1 to its child node 2.

- Such procedure continues until all nodes in the network have derived their initial data rates as shown in Figure 3.4(d).

3.4.2.2 Adaptation Phase

Every W time interval, the BS triggers a data rate adaptation round. In each round, a parent node and its children adjust their data rates in a collaborative manner to improve the utility of the neighborhood.

Figure 3.5 illustrates the behaviors of node i as a parent node. When it receives an ACK from parent node p, node i extracts $\Delta_{p,i}$ whose value can be δ, $-\delta$ or zero. δ is a system parameter. $\Delta_{p,i} = \delta$ means that the outgoing data rate of the subtree rooted at node i (denoted as T_i) can be increased by δ without violating the sustainability requirement at node p. $\Delta_{p,i} = -\delta$ means that the subtree's outgoing data rate shall be reduced by δ; otherwise, node p will be overloaded. $\Delta_{p,i} = 0$ means the subtree's outgoing data rate shall not increase. Meanwhile, node i estimates its residual energy at the end of the charging round as

$$\hat{e}_i = e_i - \omega(e_{tx} r_i^{out} + e_{rx} r_i^{in} + e_{sx} r_i) + \omega \hat{H}_i + \hat{A}_i, \tag{3.15}$$

where r_i^{out} and r_i^{in} are the measured outgoing and incoming data rate at node i, respectively. ω is the rest time length of the current charging round and $\hat{A}_i$ is the charging energy node i will received during

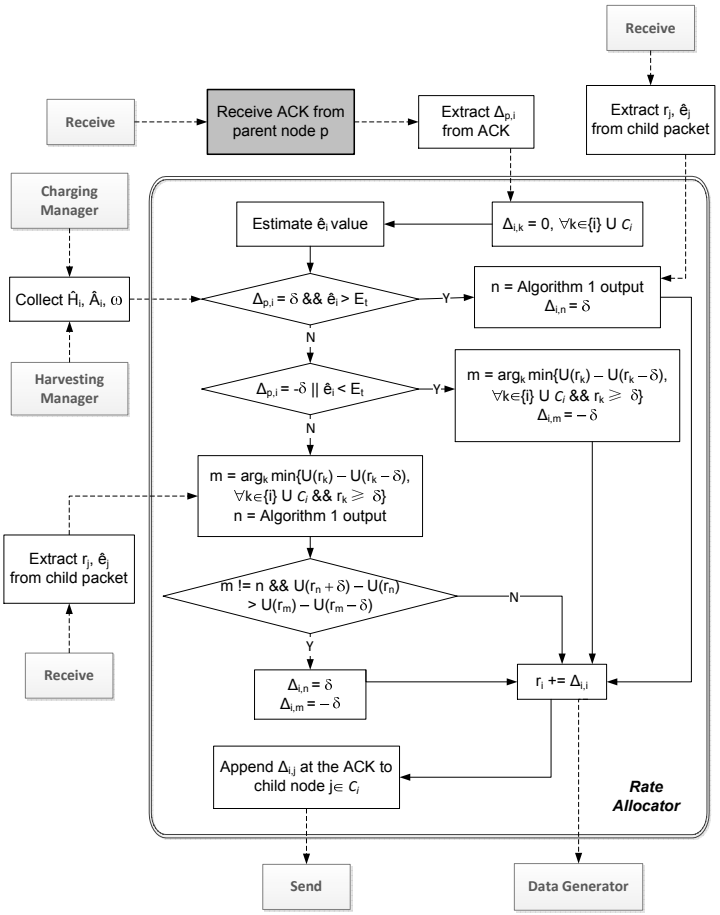


Figure 3.5 Flow chart of data rate adaptation when node i acts as parent.

ω time. Depend on $\Delta_{p,i}$ and $\hat{e}_i$ values, node i behaves differently as follows:

- $\Delta_{p,i} = \delta$ and $\hat{e}_i > E_t$. In this case, both nodes p and i have extra energy to allow more data generation in T_i, which means that higher network utility can be achieved. To maximize the utility improvement, node i runs Algorithm 3 to select a node, say n, among i and its child nodes such that increasing its data rate by δ will result in the maximum utility gain while the sustainability requirement in the neighborhood is still maintained. If node i is selected, r_i is increased by δ. Otherwise, node i sets $\Delta_{i,n} = \delta$. Then, $\Delta_{i,j}, \forall j \in C_i$ is appended to the ACK for node j. For child node $j \neq n$, $\Delta_{i,j}$ value is zero.

- $\Delta_{p,i} = -\delta$ or $\hat{e}_i < E_t$. In this case, either node p or node i has been overused. In either case, the outgoing data rate of T_i shall be reduced. To minimize the utility loss due to data rate reduction,

node i selects a node, say m, among i and its child nodes such that decreasing its data rate by δ will result in the minimum utility loss. If node i is selected, it decreases r_i by δ. Otherwise, $\Delta_{i,m}$ is set to $-\delta$.

- If neither Case I nor II is satisfied, it means that the outgoing data rate of T_i cannot be further increase and it does not have to be decreased. In this case, node i ties to improve the utility in the neighborhood by shifting the data rates among itself and its child nodes while ensuring the outgoing data rate of T_i is not changed. Particularly, node i picks a node, say n, increasing whose data rate by δ results in the maximum utility gain and picks another node, say m, decreasing whose data rate by δ results in the minimum utility loss. If the utility gain is larger than the utility loss, node i then sets $\Delta_{i,n} = \delta$, $\Delta_{i,m} = -\delta$ and adjusts the r_i value accordingly if m or n equals i. Otherwise, $\Delta_{i,j} = 0, \forall j \in C_i$ and r_i is unchanged.

When child node j receives the ACK from parent node i, it also runs the adaptation algorithm as node i does. Hence, the data rate adaptation procedure propagates from the BS to leaf nodes in each round. When j's data rate is low, it may take a while for node i to receive a data packet from j and then append the $\Delta_{i,j}$ value in the ACK. To handle this case, the ACK packet is utilized in the best-effort manner. After adjustment, if node i has no ACK to j in a certain time interval, it will send a control message containing $\Delta_{i,j}$ value to node j. In this work, the interval is set to one minute.

As the BS has infinite energy supply, when it triggers the adaptation, it sets $\Delta_{BS,j} = \delta$ for its child node j, which means the BS always has enough energy to receive more data packets from the network.

3.5 Performance Evaluation

3.5.1 Simulation Setup

NS-2 based simulations have been conducted to evaluate the JCRA performance in terms of *network utility*. The changing trace of the minimal nodal energy in the network is also plotted to verify the sustainability requirement. In the simulation, we set $U(r_i)$ in Equation 3.1 as $1 - e^{-r_i}$ and set $\eta_i(s)$ in Equation 3.2 as $0.0958 * d^2(i,s) 0.0377 d(i,s) + 1.0$ according to [38]. We compare JCRA with the

Algorithm 3 Node selection for data rate increase

Input: $\{\hat{e}_j, r_j, j \in C_i\}, \hat{H}_i, \hat{A}_i, \omega$
Output:node id that should increase its data rate

Compute $\hat{e}_i$ with Equation (3.15)
$n \leftarrow -1, U^* \leftarrow 0$
/* Check if can increase r_i by δ */
if $\hat{e}_i - (e_{tx} + e_{sx})\delta\omega > E_t$ **then**
  $n \leftarrow i$
  $U^* \leftarrow U(r_i + \delta) - U(r_i)$
/* Check if i can relay extra δ data */
if $\hat{e}_i - (e_{tx} + e_{rx})\delta\omega < E_t$ **then**
  return n
for each child node $j \in C_i$ **do**
  if $\hat{e}_j - (e_{tx} + e_{sx})\delta\omega > E_t \&\& U(r_j + \delta) - U(r_j) > U^*$ **then**
    $n \leftarrow j$
    $U^* \leftarrow U(r_j + \delta) - U(r_j)$
return n

centralized, off-line Algorithm 2, which is denoted as OPT. For both JCRA and OPT, the average data rate over the simulation time is used to compute the network utility in Equation 3.1.

In the simulation, the deployment field is a 400 m $\times$ 400 m area and the BS is located at the center. The trajectory is a closed curve which starts from and ends at the BS. A designated percentage of nodes are randomly deployed within the distance D_{max} of the trajectory (called near-trajectory nodes). The rest of nodes are randomly deployed in the area. Wireless charging efficiency model in [38] is employed in the simulation and the charging output power $\Lambda_c = 3W$. The topology is connected and the CTP protocol [26] is employed to build the initial data collection tree. Solar is the ambient energy source. Each nodes is equipped with a historical solar profile shown in Figure 3.6. The actual solar energy for each node is a random variant of the equipped historical profile.

The evaluation results are averaged over 30 different random topologies. We vary the targeted energy E_t, charging round time length τ, MC rescheduling interval I_{MC}, the network density and the percentage of near-trajectory nodes in the simulation. The node battery capacity is 10000 J. The initial nodal energy equals E_t. The maximal communication range is 70 meters and RI-MAC [50] is used as the underlying MAC protocols. Based on the default RI-MAC settings and the energy model in [51], $e_{tx} = 0.05$ J/pkt, $e_{rx} = e_{sx} = 0.02$ J/pkt. Note that the actual packet transmission cost in the

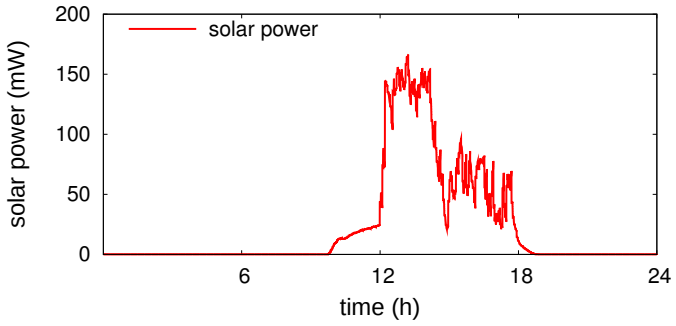


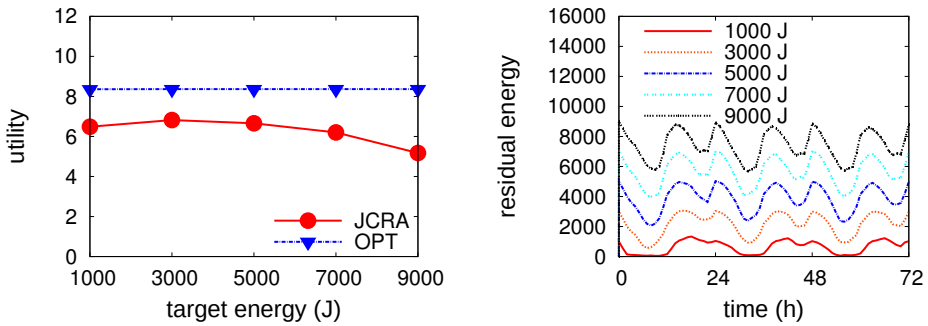
Figure 3.6 Historical solar power profile.

simulation may be larger than 0.05 J/pkt due to channel deterioration or interference. The BS triggers an adaptation round every five minutes. $\delta = 0.05$ pkt/s and the simulation lasts for 72 hours.

3.5.2 Simulation Results

3.5.2.1 Network utility with varying E_t

Figure 3.7 compares the performances of JCRA and OPT with the E_t values varying between 1000 J and 9000 J. $\tau = 12$ h, $I_{MC} = 3$ h, number of nodes in the network is 60 and 50% nodes are near-trajectory nodes. Figure 3.7(b) plots changing trace of the minimal nodal energy in the network. Figure 3.7(a) shows that JCRA achieves maximum network utility when $E_t = 3000$ J. As mentioned



(a) Network utility.

(b) Minimum nodal energy.

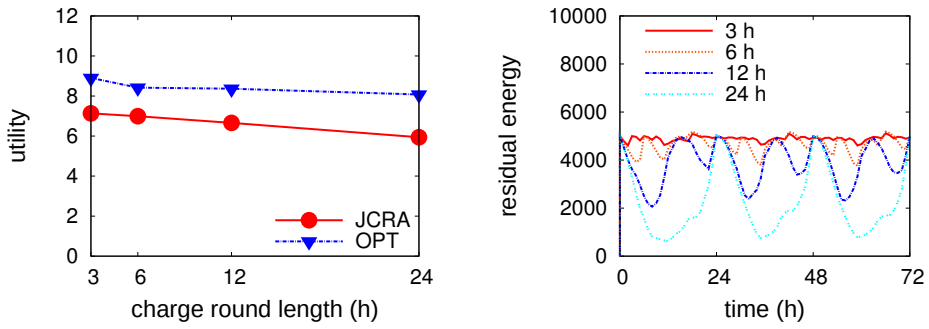
Figure 3.7 Performance comparison under different E_t.

before, the amount of E_t energy serves as the energy buffer to tolerate environmental and network uncertainties. When the energy buffer is small, i.e., $E_t = 1000$ J, some nodes may run out of energy when the actual solar harvesting rate is lower than estimated or the MC charging distance is farther than expected due to inaccurate robot localization, as illustrated in Figure 3.7(b). When a node runs out of energy, all data packets generated at its subtree cannot reach the BS, resulting the degraded network utility. On the other hand, when the energy buffer is large, $E_t = 9000$ J, some nodes' energy may hit the battery ceiling when the actual solar harvesting rate or the wireless charging rate is higher than expected. And the supplied energy will not be fully utilized. Therefore, E_t shall be set to a value around half of the battery ceiling. For example, when $E_t = 3000$ and $E_t = 5000$ J, JCRA achieves 86% and 85% of the OPT, respectively. Since the OPT scheme is an off-line algorithm with perfect channel conditions, the achieved network utility is not affected by E_t.

Figure 3.7(b) verifies that the network sustainability requirement is satisfied when $E_t > 1000$ J as the minimal nodal energy returns to E_t value at the end of each charging round, e.g., every 12 hours.

3.5.2.2 Network utility with varying τ

We now evaluate the impact of the charging round time length τ. Results are shown in Figure 3.8.



(a) Network utility.

(b) Minimum nodal energy.

Figure 3.8 Performance comparison under different τ.

From Figure jcra:fig:charge-time(a), we can see that as the charging round time length τ increases, the network utility achieved by both JCRA and OPT decreases. This is because, to satisfy the sus-

tainability requirement, the estimated nodal energy at the end of the charing round $\hat{e}_i$ shall be no less than E_t. As a node may have inaccurate estimation about future solar energy harvesting rate and other network conditions, when τ value increases, $\hat{e}_i$ value deviates farther away from its actual value. When $\hat{e}_i$ is smaller than the actual value, a node will be reluctant to increase its data rate and the possibility to hitting the battery ceiling increases. Similarly, when $\hat{e}_i$ is larger than the actual value, a node will be aggressive to increase the data rate and more likely this node will run out of energy during the simulation. In either case, the achieved network utility will decrease. This statement can be verified by Figure 3.8(b). As we can see, when the charging round time length increases, the minimum nodal energy variance also increases. Note that Figure jcra:fig:charge-time(b) is the average over 30 random topologies. The increased variance means that the minimum nodal energy will hit zero more frequently as τ increases.

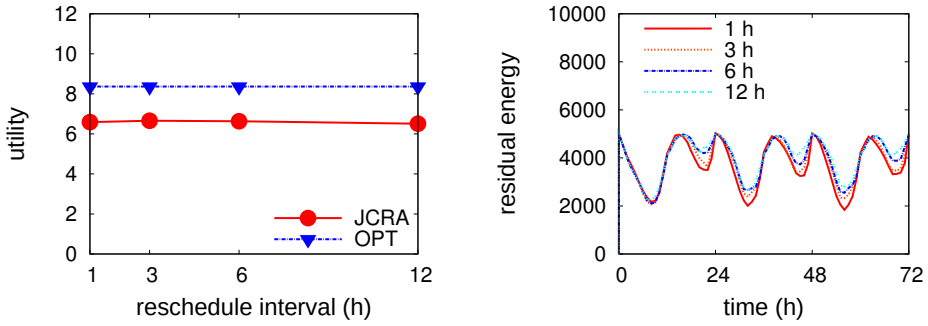
As the average solar rate over a charging round $\hat{H}_i$ is use in OPT, the longer the charing round is, the less accurate that $\hat{H}_i$ represents the actual solar power, and the performance of OPT degrades.

3.5.2.3 Network utility with varying I_{MC}

Figure 3.9 shows how the MC rescheduling interval I_{MC} affects the network utility. $E_t = 5000$ J, $\tau = 12$ h, number of nodes in the network is 60 and 50% nodes are near-trajectory nodes. Since the charging activities in OPT is not rescheduled in a charing round, the network utility achieved by OPT does not change when I_{MC} varies. Different I_{MC} values also have little impact on the JCRA performance. This is because the bottleneck nodes on the collection tree are relatively stable. Even the MC reschedules the charging activities more frequently, the new schedule will not be significantly different from the previous one. At different I_{MC} values, the network utility achieved by JCRA is around 83% of the one achieved by OPT.

3.5.2.4 Network utility with varying network density

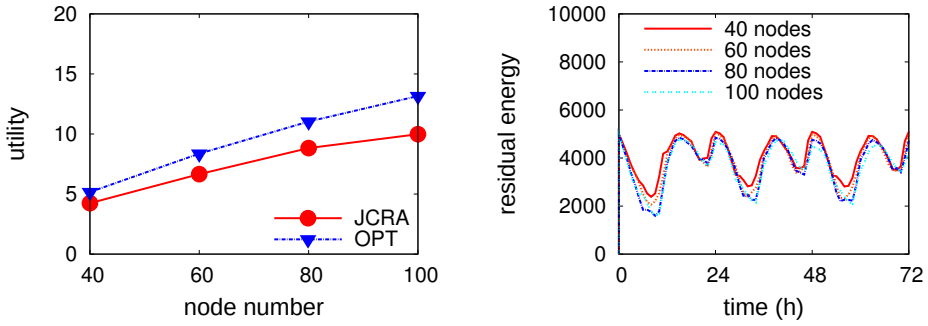
The impact of network density on the JCRA performance is shown in Figure 3.10. $E_t = 5000$ J, $\tau = 12$ h, $I_{MC} = 3$ h and 50% nodes are near-trajectory nodes. As more and more sensor nodes are deployed to generate data, the network utility achieved by both OPT and JCRA increases. Even with



(a) Network utility. (b) Minimum nodal energy.

Figure 3.9 Performance comparison under different I_{MC}.

100 nodes deployed, JCRA achieves 78% of the OPT's network utility.



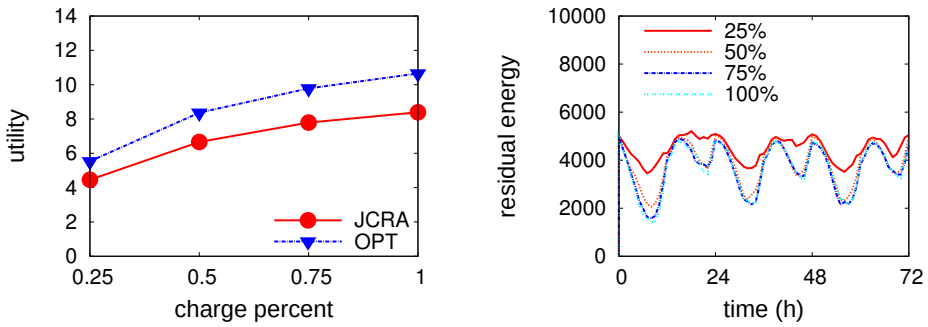
(a) Network utility. (b) Minimum nodal energy.

Figure 3.10 Performance comparison under different network sizes.

3.5.2.5 Network utility with varying percentage of near-trajectory nodes

Figure 3.11 compares the performances of JCRA and OPT when the percentage of the near-trajectory nodes varies from 25% to 100%. $E_t = 5000$ J, $\tau = 12$ h, $I_{MC} = 3$ h and number of nodes in the network is 60. As more and more nodes are deployed near the trajectory, the MC has more and more control over the energy distribution and hence the data rate generation of the network. There-

fore, the achieved network utility by both JCRA and OPT increases. Regardless of the near-trajectory nodes percentage, the network utility achieved by JCRA is around 80% of OPT.



(a) Network utility.

(b) Minimum nodal energy.

Figure 3.11 Performance comparison under different percentage of near-trajectory nodes.

3.6 Conclusions

In this study, we propose a practical and efficient joint charging and rate allocation scheme, called JCRA, to maximize the network utility while ensuring eternal network lifetime. We present the design and implementation of the JCRA scheme and show its effectiveness in improving the network utility compared with the centralized, off-line optimal solution via ns-2 simulations, under various configurations.

In the future, we will improve the design by taking routing strategy into account. As routing behaviors shape the network traffic distribution, a joint charing, routing and rate allocation design may improve the network utility even further.

CHAPTER 4. LIFETIME BALANCED DATA AGGREGATION IN DELAY-BOUNDED ENERGY-HETEROGENEOUS SENSOR NETWORKS

4.1 Introduction

4.1.1 Motivations

In a sensor network, sensor nodes are usually powered by small batteries with limited energy supplies. When applying the network for long-time applications such as continuous environmental monitoring, how to prolong the network lifetime is of critical importance.

Through eliminating inherent redundancy in raw sensory data, in-network data aggregation [52,53] has been widely applied as an effective technique to reduce communication cost and extend the lifetime of a sensor network. With the data aggregation mechanism, a node should be allowed to hold data received or generated by itself for a while, aggregate the data in bulk, and send out only the aggregated results. The extend to which data volume can be suppressed highly depends on how long a node can hold data before sending them out. Generally, the longer can a node hold data, the more data can it suppress and hence the higher is the communication efficiency. However, holding data introduces extra data delivery delay. In many sensor network applications, the value of sensory data could be greatly depreciated or even become zero if the data is delivered to the sink with a delay longer than a certain application-specific delay bound. Therefore, the allowed holding time is constrained by the application-specific requirement on end-to-end data delivery delay.

It is important to make full use of the available but constrained holding time for aggregation to prolong network lifetime. This becomes especially demanding in the context of multi-hop sensor networks. Along each multi-hop source-to-sink path, the allowed holding time should be allocated to all nodes appropriately to ensure the required end-to-end data delivery delay is not violated along the

path. Research [54] has been conducted to optimize the distribution of holding time according to the distribution of data traffic and the network topology. However, non-uniform nodal lifetime, which is a critical factor impacting network lifetime, has been neglected.

Due to various environment and system reasons, sensor nodes in the same network may have various nodal lifetime. For example, nodes with batteries of poorer quality, nodes that are bottlenecks on a collection tree, and solar-rechargeable nodes deployed to shady locales may have shorter lifetime than their peers. As the energy depletion in a sensor node may cause network disconnection or create coverage holes, which could render the entire sensor network nonfunctional, many sensor network applications [6, 55, 56] define the network lifetime as the minimal nodal lifetime among all sensor nodes in the network. Therefore, in order to prolong the network lifetime, it is critical to prolong the lifetime of the shortest-nodal-lifetime nodes.

Despite the need for a multi-layer holistic approach to balance nodal lifetime and thus prolong the network lifetime, it is necessary to design a data aggregation scheme that can take into account the non-uniform nodal lifetime among nodes, attempt to balance their nodal lifetime, and thus more effectively prolong the network lifetime.

4.1.2 Literature Survey

Many in-network data aggregation protocols [57–59] have been proposed, but the timeliness of the data delivery was not a concern. Although Ye *et al.* [60] formulated the energy-delay tradeoff problem as a semi-Markov decision process by depreciating the data revenue as aggregation holding time increases, no explicit end-to-end delivery delay bound was considered. To bound end-to-end delivery delay, many existing works [61–63] require time synchronization between neighboring nodes. Solis *et al.* [61] employed the concept of cascading timeout where a node's aggregation timeout happens right before its parent's to achieve high aggregation degree with small delay overhead. Xiang *et al.* [62] explored the joint data aggregation and timeliness of data delivery problem, and proposed a utility-based scheme called tPack to minimize the whole network communication cost. Assuming a synchronized time-slotted system, [63] formulated the energy-delay tradeoff problem as an integer optimization problem. Different from these works, our scheme does not require synchronization. Moreover, all of the

afore-mentioned works aim to minimize the energy consumption, without considering the lifetime balance between nodes which is critical in improving the network lifetime.

[54] investigated the problem of energy efficient data delivery within a delay bound and proposed two distributed schemes to balance energy consumption among sensor nodes. However, the schemes work only in homogeneous networks (e.g., the battery quality, radio energy consumption rate and initial nodal energy are the same) whereas our scheme can deal with the heterogenous situations. There have also been works [64, 65] on optimizing aggregation tree structure. Our scheme is orthogonal to these works, because our scheme works under any aggregation tree structure.

4.1.3 Contributions

In this study, we present LBA, a low cost, asynchronous, delay-constrained data aggregation scheme for duty cycle sensor networks. LBA aims to prolong the network lifetime. The key idea is to dynamically adjust the aggregation holding time between two neighboring nodes and hence to balance their nodal lifetime. As neighboring nodes keep balancing their nodal lifetime, the nodal lifetime of all nodes in the entire network can be balanced gradually and the network lifetime can be extended.

We have implemented and experimented LBA in a testbed of 32 TelosB sensor nodes. Experimental results show that LBA effectively achieves the design goal of balancing the nodal lifetime, and prolongs the sensor network lifetime under various network configurations, especially the heterogeneous ones. Through theoretical analysis, we also proposed a network lifetime upperbound. According to our evaluation results, LBA can approach the lifetime upperbound especially when the nodes have highly different nodal lifetime.

4.2 System Model and Problem Statement

4.2.1 System Model

We study data aggregation in a sensor network, where each sensor node could periodically generate and report sensory data, and all the nodes form a collection tree rooted at the sink. The nodes may not be time-synchronized or energy-synchronized (i.e., their energy supplies and consumption rates may not be uniform). The nodes are duty-cycled for energy saving, which are typical when the network is

deployed for long-time monitoring. To build and maintain the data collection tree, a routing protocol such as the collection tree protocol (CTP) [26] may be employed. Underlying the routing protocol, an asynchronous and duty cycle MAC protocol such as RI-MAC [50] may be adopted.

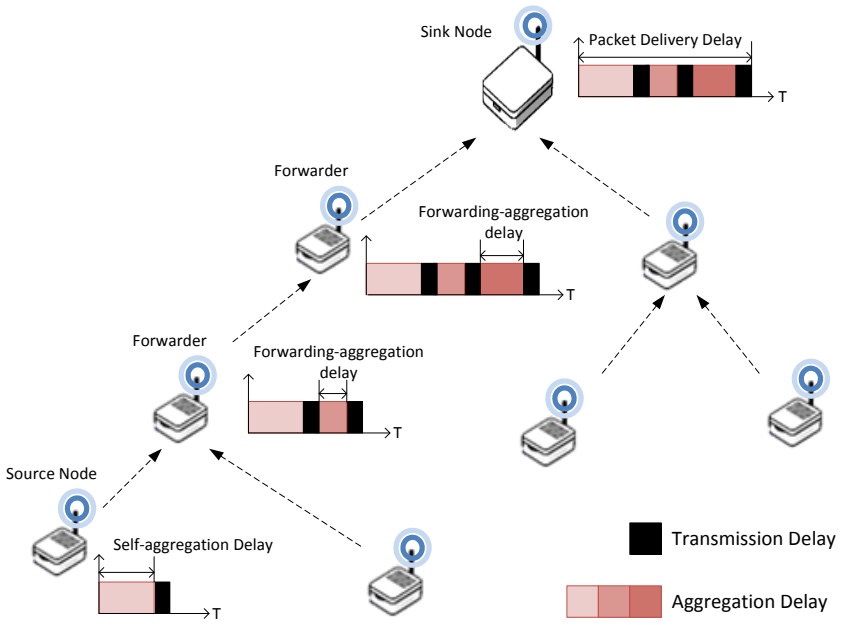


Figure 4.1 System model.

A typical data aggregation process is shown in Figure 4.1. A source node may not send out sensory data packet immediately after it is generated; instead, the node may wait a certain period of time (*self-aggregation delay*), attempting to aggregate multiple data packets generated during the period and thus reduce the amount of data sent to its parent node. Similarly, a forwarding node may not forward a data packet immediately after the reception. It may wait for another period of time (*forwarding-aggregation delay*) in order to aggregate multiple data packets received during the period. Consequently, the delay involved in delivering a data packet from source to sink is composed of a self-aggregation delay and multiple forwarding-aggregation delays. In the rest of this study, we denote the average data generation rate at each node i as λ_i, and the average forwarding-aggregation and self-aggregation delay at node i as w_i and w_i' respectively.

We assume the *total aggregation* model [66] in data aggregation; that is, an arbitrary number of data packets that are available at the aggregation time can be suppressed into a single data packet. Such model can be seen in many sensor network applications. For example, in monitoring applications, users

often are more interested in the maximum, minimum or average values, or the percentile statistics, of sensory data, rather than the raw data themselves.

As we can see, the longer is the aggregation delay, the more data packets can be aggregated and thus the higher is the communication efficiency. However, monitoring applications often also require that data delivery delay be lower than a certain bound to assure timely awareness of the monitored environment. We assume the following generic delay requirement: *at least p percent of sensory data should be delivered to the sink within time D after the data has been generated, where D and p are application-specific parameters.*

4.2.2 Problem Statement

Our design objective is to dynamically determine the forwarding-aggregation and self-aggregation delays (i.e., w_i and w_i') for each node i to try to balance the lifetime of sensor nodes and hence prolong the network lifetime, under the condition that the data delivery delay requirement is satisfied. In this course, the differences between nodes, for example, different nodal energy levels, energy consumption rates and data generation rates, should be considered. Specifically, the problem can be formalized as follows:

Given:

$$T,\ \{\lambda_i\}, \{e_i\}.$$

Objective:

$$\max \min\{L_i\}$$

Subject to:

$$\forall l, w_l' + \sum_{i \in s_l} w_i \leq D_p, \text{where } l \text{ is leaf}$$

Output:

$$\{w_i, w_i'\}.$$

Here, T is the collection tree topology, e_i is the residual energy and L_i represents the nodal lifetime of node i. l is a leaf node and s_l denotes the path connecting the parent of l and the sink in T. D_p represents the maximal allowed delay, including both transmission and aggregation delay, along each

source-sink path such that at least p percent of the source's sensory data can reach the sink within D time. The calculation of L_i and D_p is to be presented in Section 4.4.

4.3 Design Overview

Solving the above optimization problem in a centralized manner is impractical as it requires each node to know the residual energy levels, data generation rates of all other nodes, and the topology of the network. Acquiring these information could incur high communication overhead because of potentially large network scale and dynamic nature of the information. Therefore, we approach the problem in a distributed and localized manner. Specifically, coordinations only take place between neighboring parent-child nodes, which exchange information with each other and coordinately adjust their aggregation delays.

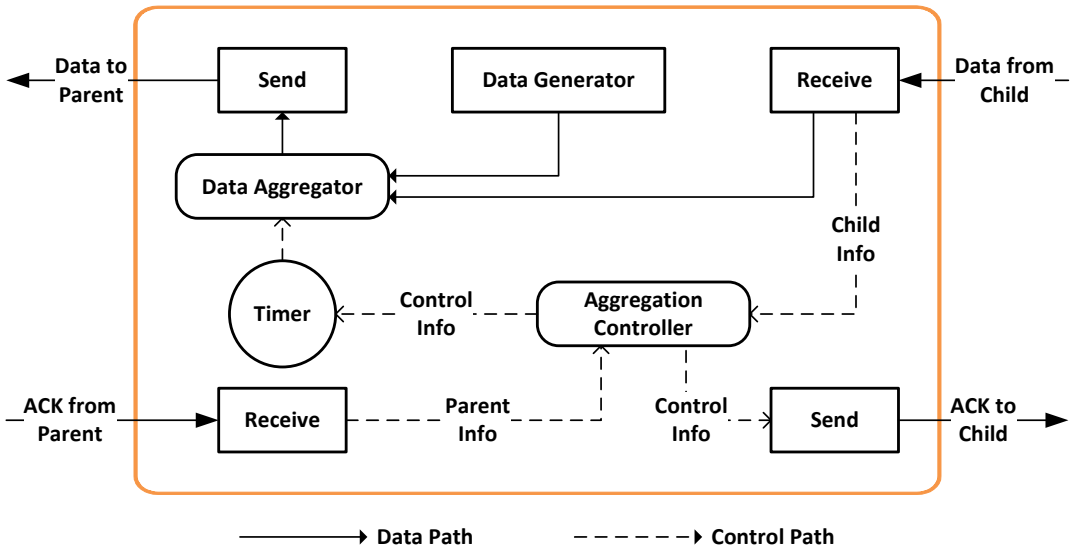


Figure 4.2 Design overview.

As shown in Figure 4.2, the coordination is realized by piggy-backing some small pieces of control information at the end of the data and ACK packets exchanged between parent-child nodes. Specifically, when sending out a data packet to its parent, a node appends to the packet its estimated nodal lifetime and some other information related to the adjustment of aggregation delays. Upon receiving the packet, the parent node pushes the packet into a queue while feeding the appended information to the "Aggregation Controller" module. Based on the information, the parent node adjusts its own w_i,

and appends some information to the ACK to instruct its children nodes on how to adjust their w_i values. When the ACK reaches a child node j, the node changes its w_j and w'_j accordingly. This way, as every parent-child pair keeps attempting to balance their nodal lifetime, the nodal lifetime of all nodes in the entire network is also adjusted gradually towards the balanced status.

4.4 Analytical Study

To provide a theoretical foundation to direct and evaluate the performance of our design, we present extensive analysis in this section. Specifically, we first provide a formulation of nodal lifetime as a function of a node's data input/output rates. Then, the data I/O rates are formulated as functions of nodal data generation rates, self-aggregation and forwarding-aggregation delays. This is followed by the computation of the maximum aggregation and transmission delays that all the nodes on a source-sink path are allowed to introduce without violating a certain application-specific data delivery delay requirement. Based on the above analysis, we finally propose an algorithm to compute an upperbound for the network lifetime through intelligently determining the self-aggregation and forwarding-aggregation delays for each node.

4.4.1 Nodal Lifetime

As our design targets to be applicable in time-asynchronous duty cycle sensor networks, a time-asynchronous duty cycle MAC protocol is assumed. In the analysis and detailed design presented in this study, we take RI-MAC [50], a well-known asynchronous and duty-cycled MAC protocol, as an example. Note that, our design can work with other MAC protocols, e.g., X-MAC [33], A-MAC [67], after replacing the nodal lifetime analysis module of RI-MAC with that of the alternative MAC protocol.

For self-containedness, the sketch of RI-MAC is illustrated in Figure 4.3 where T_r is the receiver's beacon interval, ϕ is the receiver's channel checking period and τ is the time needed by the sender and receiver to exchange a data packet and the corresponding ACK. In this protocol, each node periodically wakes up for time ϕ every interval T_r to check if there is any incoming packet intended for the node. After turning on its radio, a node immediately broadcasts a beacon, announcing its readiness for receiving packets. If a node has packet to send (e.g., node S in Figure 4.3) it turns on its radio if it is off,

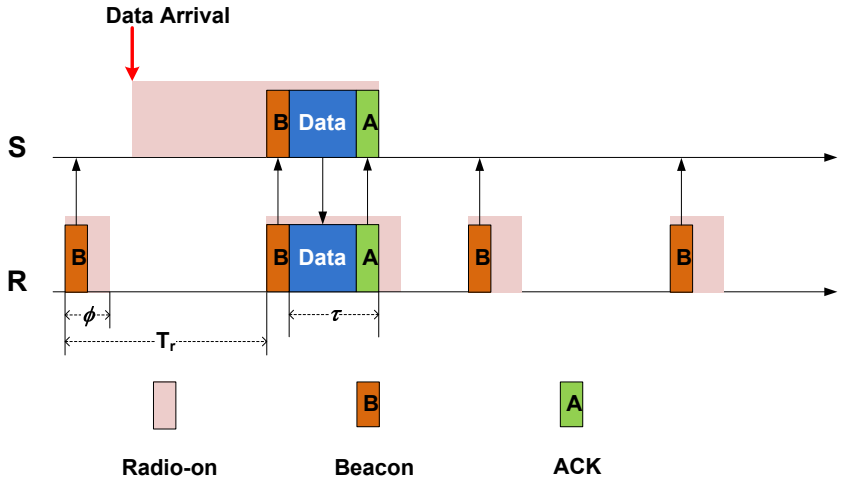


Figure 4.3 The sketch of RI-MAC protocol.

and keeps the radio on to wait for the beacon from the intended receiver (e.g., node R in Figure 4.3). Upon receiving the expected beacon, S sends out its data immediately, which will be acknowledged by R with an ACK after R has received the data. Then, S turns off its radio, while R and any other sender who has pending packet intended for R can start transmission. If there are no more incoming/outgoing packets, R turns off its radio and goes to sleep. Further, due to the lossy nature of the wireless channels, a sender may need several transmission attempts to successfully send out a packet. Let ETX_i denote the expected transmission attempts[1] at node i. Let m denote the maximum number of transmission trials before a sender gives up its transmission attempt until the next time it receives beacon from its intended receiver. Then, we can estimate the nodal lifetime L_i as follows:

$$\frac{e_i}{((T_r(\lceil \frac{ETX_i}{m} \rceil - \frac{1}{2}) + \tau(ETX_i \% m))\lambda'_i + \frac{\phi}{T_r} + \tau \sum_{j \in C_i} \lambda'_j)P}. \tag{4.1}$$

Here, P is the energy consumption rate when a node's radio is on, λ'_i denotes the data output rate of node i, and τ is the time to send or receive a data packet. $\sum_{j \in C_i} \lambda'_j$ denotes the data input rate at node i, where C_i is the set of i's children and λ'_j is the data output rate of node j. A node could be both a sender and a receiver. As a sender, the node waits $\frac{T_r}{2}$ on average for its receiver to wake up and then consumes time τ for each transmission attempt until the number of unsuccessful attempts

[1]Note that routing protocols such as CTP [26] in sensor networks have provide mechanisms for automatical measuring ETX_i for node i. We therefore do not describe how to measure ETX_i in this study.

reaches m, in which case it waits for another T_r and then repeats the transmission attempts. Hence, with average number of attempts ETX_i for each data packet, the overall energy consumption for the successful transmission of a data packet is $T_r(\lceil \frac{ETX_i}{m} \rceil - \frac{1}{2}) + \tau(ETX_i \% m)$. As a receiver, the node wakes up for time ϕ every interval T_r, and spends time $\tau \sum_{j \in C_i} \lambda'_j$ on receiving data packets. Hence, the denominator of Eq. (4.1) estimates the energy consumption rate at node i. Note that, we here only estimate the energy consumption for communication, which is usually the most significant part of nodal energy consumption. It can be easily extended to include the energy consumption for other reasons such as sensing and computation.

4.4.2 Nodal Data Input/Output Rates

To facilitate theoretical analysis and evaluation of our design, we assume sensory data packets are generated at node i with the intervals following an exponential distribution of mean $1/\lambda_i$, and therefore the flow of sensory data packets generated by a node follows a Poisson distribution of mean λ_i.

Our design also intentionally shapes the data packet flows after aggregations to follow the Poisson distribution. Particularly, if node i is a leaf node, it maintains a timer A'_i which fires every time interval W'_i, where W'_i is a random variable following the exponential distribution with mean w'_i. If node i is a non-leaf node, it maintains a timer A_i which fires every time interval W_i, where W_i is a random variable following the exponential distribution with mean w_i. Every time when the timer (A'_i or A_i) fires, data packets are aggregated into a single packet and then sent to its parent node. If there is no packet to send when the timer fires, a dummy packet is sent to ensure that the output data flow follows Poisson distribution. The enhancement described later reduces dummy packets as much as possible.

Based on the above assumption and mechanism, we next formulate the nodal average data output rate (denoted as λ'_i for each node i) as a function of its data input rate (i.e., $\sum_{j \in C_i} \lambda'_j$ where C_i is the set of children nodes of i), its own data generation rate (i.e., λ_i) and its own forwarding-aggregation and self-aggregation delays (i.e., w_i and w'_i). Note that a node's data input rate is the sum of the data output rates of its children; hence, we do not need separate analysis of data input rate.

When a node is a pure source, its self-aggregation delay w'_i (i.e., the maximal time it can wait to

aggregate its own sensory data without violating the delay requirement) can be computed as follow:

$$w_i' = D_p - \sum_{k \in s_i} (w_k + d_k) - d_i, \tag{4.2}$$

where d_i denotes the average transmission delay at node i. That is, the sum of aggregation and transmission delays of all nodes on the path from node i to the sink should not exceed D_p, the maximum aggregation and transmission delay that any source-sink path is allowed to introduce. The output rate of node i can simply be $\frac{1}{w_i'}$ as one packet (aggregated data or dummy) is sent every time timer A_i' fires. However, if the average data generation interval $\frac{1}{\lambda_i}$ is greater than w_i', a large number of dummy packets may be generated; To reduce the bandwidth waste, we instead allow the data packet to be sent out immediately after its generation. With the above enhancement, the data output rate of node i, denoted as λ_i^s, can be calculated as follows:

$$\lambda_i^s = \begin{cases} \lambda_i & : w_i' \leq \frac{1}{\lambda_i}; \\ \frac{1}{w_i'} & : otherwise. \end{cases} \tag{4.3}$$

Generally, when node i is a pure forwarder or both forwarder and source, it can use timer A_i by default to regulate output data flow, and the output data rate would be $\frac{1}{w_i}$. However, when the data packets are received or generated by the node in an interval greater than w_i, many dummy packets will have to be sent. To save bandwidth, the following optimization can be applied:

- If packets from descendant nodes arrive at node i with an average interval greater than w_i, timer A_i is not needed. In this case, there are following three subcases:
  - If the arrival interval of packets from descendants is no greater than w_i', then the packets can be sent in the following way. Whenever a packet from descendants arrives at i, it is aggregated with any un-sent packets generated by i, and the aggregation result is sent to the parent node.
  - If the arrival interval of descent packets is greater than w_i' and node i's self-generated packets arrive at an interval no smaller than w_i', then any packet (no matter it is from descendants or self-generated) is forwarded to the parent node whenever it arrives at i.

– Otherwise, timer A'_i is set to fire every interval W'_i, where W'_i is a random variable following the exponential distribution of mean w'_i. Packet aggregation and transmission rules are as follows. Whenever a descendant packet arrives at i, it is aggregated with any un-sent self-generated packets, and the aggregation result is then sent to the parent node. Whenever timer A'_i fires, all packets held at node i are aggregated and the aggregation result is sent to the parent node; if there is no un-sent packet at i, a dummy packet is sent to the parent.

- Otherwise, timer A_i is used, and the default way for packet aggregation and sending is applied.

To summarize, node i's data output rate λ'_i can be calculated as follows:

$$\lambda'_i = \begin{cases} \sum_{j \in C_i} \lambda'_j & : w_i \leq \frac{1}{\sum_{j \in C_i} \lambda'_j} \leq w'_i \\ \lambda_i + \sum_{j \in C_i} \lambda'_j & : w'_i \leq \frac{1}{\lambda_i}, w'_i \leq \frac{1}{\sum_{j \in C_i} \lambda'_j} \\ \frac{1}{w'_i} + \sum_{j \in C_i} \lambda'_j & : w'_i > \frac{1}{\lambda_i}, w'_i \leq \frac{1}{\sum_{j \in C_i} \lambda'_j} \\ \frac{1}{w_i} & : otherwise. \end{cases} \tag{4.4}$$

With the above optimization, the data output flow from i still follows the Poisson distribution.

4.4.3 Lowerbound of a Subtree's Data Output Rate

Based on the above analysis of nodal data output rates, we next present a lowerbound of a subtree's data output rate in the following Lemma 4.4.1. Note that the result is also to be used in computing the performance upperbound.

Lemma 4.4.1 *Let T_i denote a subtree rooted at i, and w_i and w'_i be the forwarding-aggregation and self-aggregation delays of node i respectively. Then a lowerbound of the data output rate of T_i, denoted as $\hat{\lambda}'_i$ is as follows.*

$$\hat{\lambda}'_i = \begin{cases} \sum_{j \in T_i} \lambda_j & : w'_i \leq \frac{1}{\sum_{j \in T_i} \lambda_j}; \\ \frac{1}{w'_i} & : otherwise. \end{cases} \tag{4.5}$$

proof: 2 *(sketch) There are totally two cases as in the following.*

Case I: $w_i' \leq \frac{1}{\sum_{j\in T_i}\lambda_j}$. *For this case, we prove by contradiction that the data output rate of* T_i *cannot be less than* $\sum_{j\in T_i}\lambda_j$. *Suppose the output rate is lower than the value, then some data should have been suppressed at some node. Suppose node* $k \in T_i$ *is such a suppressing node but none of its descendants can suppress data. The overall data generation rate at the subtree rooted at* T_k *is* $\sum_{j\in T_k}\lambda_j$, *which is no greater than* $\sum_{j\in T_i}\lambda_j$; *i.e.,*

$$\frac{1}{\sum_{j\in T_i}\lambda_j} \leq \frac{1}{\sum_{j\in T_k}\lambda_j}. \tag{4.6}$$

Also due to $w_k \leq w_i'$, $w_k' \leq w_i'$ *and* $w_i' \leq \frac{1}{\sum_{j\in T_i}\lambda_j}$, *it holds that*

$$w_k \leq max\{w_k, w_k'\} \leq \frac{1}{\sum_{j\in T_k}\lambda_j} \leq \frac{1}{\sum_{j\in C_k}\lambda_j'}. \tag{4.7}$$

Further, it is obvious that $\sum_{j\in T_k}\lambda_j \geq \lambda_k$; *together with Eq. (4.7), it holds that*

$$w_k' \leq max\{w_k, w_k'\} \leq \frac{1}{\sum_{j\in T_k}\lambda_j} \leq \frac{1}{\lambda_k}. \tag{4.8}$$

According to Equations (4.4), (4.7) and (4.8), it follows that $\lambda_k' = \lambda_k + \sum_{j\in C_k}\lambda_j'$, *where, as no descendant of* k *suppresses data,* $\sum_{j\in C_k}\lambda_j' = \sum_{j\in C_k}\sum_{l\in T_j}\lambda_l$. *Therefore,* $\lambda_k' = \sum_{j\in T_k}\lambda_j$; *i.e., the number of data packets does not decrease, which is a contradiction.*

Case II: $w_i' > \frac{1}{\sum_{j\in T_i}\lambda_j}$. *In this case, the data output rate of* T_i *is* $\frac{1}{w_i'}$ *when* w_i *is set to* w_i'; *that is, totally* $A = \sum_{j\in T_i}\lambda_j - \frac{1}{w_i'}$ *amount of data is suppressed. We can show that, in other settings, the amount of suppressed data cannot exceed* A. *Specifically, suppose nodes* $j_1, j_2, \cdots, j_n$ *in* T_i *can suppress data. Then the amount of suppressed data is at most* $A' = \sum_{j\in T_i}\lambda_j - \sum_{k=1}^{n}\frac{1}{w_{j_k}}$, *according to Eq. (4.10). Also note that, for each* $k = 1, \cdots, n$, $w_{j_k} \leq w_{j_k}' \leq w_i'$; *hence,* $\sum_{k=1}^{n}\frac{1}{w_{j_k}'} \geq \frac{1}{w_i'}$. *Therefore,* $A' \leq A$.

4.4.4 Maximum Total Aggregation and Transmission Delay Allowed for Each Source-Sink Path

Suppose the application-specific data delivery requirement is that at least p percent of data should be delivered to the sink within time D after being generated. The maximum total aggregation and

transmission delay that all the nodes on a source-sink path are allowed to introduce is estimated as follows. Let

$$i_0 \to i_1 \to \cdots \to i_n \to \text{sink} \tag{4.9}$$

be a path from leaf node i_0 to the sink, and d_j be the transmission delay at node i_j. Then, the maximum total aggregation and transmission delay allowed for all nodes on the path is

$$D_p = -\frac{D - \sum_{j=0}^{n} d_j}{\ln(1 - \sqrt[n]{p})}. \tag{4.10}$$

The rationale is explained by the following Lemma 4.4.2 and its proof.

Lemma 4.4.2 *If the maximum delay allowed for a source-sink path defined in Eq. (4.9) is as Eq. (4.10) and the transmission delay at node i_j is d_j, then at least p percent of the data can arrive at the sink within time D after being generated.*

proof: 3 *For any $j = 1, \cdots, n$, let us define*

$$D_j = (D - \sum_{k=0}^{n} d_k)\frac{w_{i_j}}{w'_{i_0} + \sum_{k=1}^{n} w_{i_k}} + d_j. \tag{4.11}$$

As the forwarding-aggregation delay interval W_{i_j} at node i_j follows the exponential distribution of mean w_{i_j}, the percentage of packets experiencing delay no more than D_j at node i_j is

$$1 - e^{-\frac{D_j - d_j}{w_{i_j}}} = 1 - e^{-\frac{D - \sum_{k=0}^{n} d_k}{w'_{i_0} + \sum_{k=1}^{n} w_{i_k}}}. \tag{4.12}$$

Let

$$D_0 = (D - \sum_{k=0}^{n} d_k)\frac{w'_{i_0}}{w'_{i_0} + \sum_{k=1}^{n} w_{i_k}} + d_0. \tag{4.13}$$

Similarly, we can get the percentage of packets generated by node i_0 that experiences delay no more than D_0 at node i_0 is

$$1 - e^{-\frac{D_0 - d_0}{w'_{i_0}}} = 1 - e^{-\frac{D - \sum_{k=0}^{n} d_k}{w'_{i_0} + \sum_{k=1}^{n} w_{i_k}}}. \tag{4.14}$$

Hence, the percentage of packets generated by source i_0 and experiencing delay no more than

$\sum_{j=0}^{n} D_j = D$ *during their trips to the sink is*

$$(1 - e^{-\frac{D_0 - d_0}{w_{i_0}}}) \prod_{j=1}^{n} (1 - e^{-\frac{D_j - d_j}{w_{i_j}}}) = (1 - e^{-\frac{D - \sum_{j=0}^{n} d_j}{w'_{i_0} + \sum_{j=1}^{n} w_{i_j}}})^n, \tag{4.15}$$

according to Eq. (4.11) and (4.13).

Furthermore,

$$w'_{i_0} + \sum_{j=1}^{n} w_{i_j} \leq D_p = -\frac{D - \sum_{j=0}^{n} d_j}{\ln(1 - \sqrt[n]{p})}. \tag{4.16}$$

Combining Eq. (4.16) into Eq. (4.15), we have

$$(1 - e^{-\frac{D - \sum_{j=0}^{n} d_j}{w'_{i_0} + \sum_{j=1}^{n} w_{i_j}}})^n > (1 - e^{\ln(1 - \sqrt[n]{p})})^n = p. \tag{4.17}$$

4.4.5 Performance Upperbound

In this section, we develop an algorithm (formally presented in Algorithm 4) to compute the performance upperbound, based on the nodal lifetime model built in Sections 4.4.1 and 4.4.2, the output data rate lowerbound analyzed in Section 4.4.3, and the maximum allowed end-to-end delivery latency computed in Section 4.4.4. The algorithm adopts the binary search approach to seek the maximum nodal lifetime that can be achieved by every node through attempting all legal ways of aggregation delay distribution which does not violate the delay requirement. In the computation, we have introduced several relaxations:

- Perfect channel condition has been assumed for each link; i.e., for each node i, ETX_i is always set to 1.
- When computing the output data rate for a node i using Eq. (4.4), its data input rate is assumed to be the sum of the lowerbounds of its children subtrees' output rates, where the result presented in Lemma 4.4.1 is applied. Hence, the computed output data rate for node i is also a lowerbound.
- When computing the nodal lifetime of a node i using Eq. (4.1), its data input rate is also assumed to the sum of the lowerbounds of its children subtrees' output rates.

Algorithm 4 Computation of the Performance Upperbound

Input: $\{e_i\}$, $\{\lambda_i\}$, D_p, T
Output: network lifetime upperbound $\hat{L}$

$low \leftarrow \epsilon, up \leftarrow \infty$
/* low/up is the lower/upper bound of the network lifetime */
$target \leftarrow low$
/* target is the maximum achievable network lifetime */
calculate D_p according to Lemma 4.4.2
while $up - low > \epsilon$ **do**
  $w_{sink} \leftarrow 0, w'_{sink} \leftarrow D_p$
  $reacheable \leftarrow false$
  for each node i in the pre-order traversal of T **do**
    for w_i from 0 to $D_p - \sum_{k \in s_i} w_k$ with step σ **do**
      calculate $\hat{\lambda}'_i$ according to Eq. (4.4) and (4.5) with w_i
      if $\frac{e_i}{((\frac{T_r}{2}+\tau)\hat{\lambda}'_i + \frac{\phi}{2T_r} + \tau \sum_{j \in C_i} \lambda'_j)P} \geq target$ **then**
        $reacheable \leftarrow true$
        break
  if $reacheable = false$ **then**
    $up \leftarrow target$
    $target \leftarrow \frac{low+up}{2}$
  else
    $low \leftarrow target$
    $target \leftarrow (up = \infty)?2 * low : \frac{low+up}{2}$
return low

4.5 Detailed Design

The scheme operates in two phases: initial phase and adaption phase.

4.5.1 Initial Phase

After the collection tree has been built (i.e., the routing process has completed), each node i needs to compute the initial values of w_i and w'_i. For this purpose, our scheme requires each on-tree node i to know (i) the maximum allowed aggregation and transmission delay D_p, (ii) the average transmission delay d_i for transmitting a data packet from itself to its parent, and (iii) how many hops (H_i) it is away from its furthest descendant. Knowing these information but unaware of either the data rates or energy levels of individual nodes, our scheme intends to distribute the maximum allowed aggregation delay (i.e., D_p minus transmission delays) in a fair manner. Specifically, the initial value w_i can be

determined using the following protocol:

- Denote $\bar{D}_i$ the maximal allowed delay for the subtree rooted at node i. Hence, the sink has $\bar{D}_{sink} = D_p$ and sends it to its children.
- Upon receiving $\bar{D}_i$ from parent node i, node j acts as follows:
  - If node j is non-leaf (i.e., $H_j > 0$), it sets $w_j = \frac{\bar{D}_i}{H_j+1} - d_j$ and $w'_j = \bar{D}_i - d_j$. Then, it sets $\bar{D}_j = \bar{D}_i - \frac{\bar{D}_i}{H_j+1}$ and sends $\bar{D}_j$ to its children.
  - If j is a leaf, it sets $w_j = w'_j = \bar{D}_i - \frac{T_r}{2}$ as it does not forward any data and sets $w'_j = \bar{D}_i - \frac{T_r}{2}$.

The three pieces of information needed by each node can be obtained in the following ways.

- The sink can compute and broadcast to all on-tree nodes D_p after the tree has been built. Note that, the sink can get the average per-hop transmission delay and the length of the longest branch on the tree through querying the nodes on the tree. Using these information, together with the application-specified parameters D and p, the sink can compute D_p by using Eq. (4.10).
- For the routing and route maintenance purposes, it is typical that neighboring nodes are required to exchange beacons periodically to know the expected number of transmission needed for successful delivery of a packet over a link (i.e., ETX). With the knowledge of ETX, it is easy to estimate per-transmission delay. Particularly, if the underlying MAC protocol is RI-MAC with period T_r and the ETX of the link from node i to its parent is denoted as ETX_i, then the expected per-packet transmission delay for i is $T_r(\lceil \frac{ETX_i}{m} \rceil - \frac{1}{2}) + \tau(ETX_i \% m)$ as in Eq. (4.1), where T_r and m are parameters in RI-MAC.
- To help a non-leaf node know how many hops it is away from its furthest leaf node, a simple method is as follows. Each data packet sent from a leaf node is piggy-backed with a field H which is initiated to 0. As the packet is forwarded hop-by-hop upwards, the value in the field is incremented. By observing the H values of forwarded packets for a certain period of time, each non-leaf node i can obtain its H_i.

4.5.2 Adaption Phase

In this phase, each node interacts with its parent unless its parent is the sink; it also interacts with its children nodes unless it is a leaf node. In both types of interaction, the node and its parent or children need to adjust their forwarding-aggregation and self-aggregation delays, attempting to gradually balance their lifetime. The behaviors of a node in such interactions are different depending on whether it acts as a parent (i.e., interacting with its children) or a child (i.e., interaction with it parent). As a basic difference, a child node j needs to report to its parent its current lifetime (L_j), forwarding-aggregation delay (w_j), self-aggregation delay (w'_j), data output rate (λ'_j) and data input rate ($\sum_{k \in C_j} \lambda'_k$). To save communication overhead, these information can be piggy-backed to data packets sent to its parent. As a parent node, on the hand, needs to make decision on how to adjust the forwarding-aggregation and self-aggregation delays of its own and its children, and notify the decision to its children. More specifically, the parent and child behaviors are described as follows.

4.5.2.1 Behaviors of a Parent Node

Figure 4.4 illustrates the behaviors of node i as a parent node. When receiving a data packet from its child node j, parent node i extracts L_j, w_j, w'_j, λ'_j and $\sum_{k \in C_j} \lambda'_k$. Every a certain time interval, the parent node is triggered by a timer to start checking the lifetime difference between itself and its children, which may be followed by adjustments of the aggregation delays when necessary. Particularly, only if the lifetime difference between node i and the its shortest-lifetime child is greater than a certain threshold α, the adjustments are performed. The timer interval and the threshold α are both adjustable system parameter that should be appropriately set, not too large or too small, in order to not miss the opportunity of lifetime balance or introduce unnecessary trashes.

When it is necessary to adjust aggregation delays, there are two cases as follows.

- *Case I*: Node i has longer lifetime than its shortest-lifetime child (i.e., $L_i > \min_{j \in C_i} L_j + \alpha$). In this case, the scheme will attempt to increase the lifetime of the shortest-lifetime child. Letting $\Delta = \min(\delta, w_i)$, where δ is a small value, the following adjustments are applied.
  - For node i: $w_i = w_i - \Delta$ and w'_i remains unchanged.

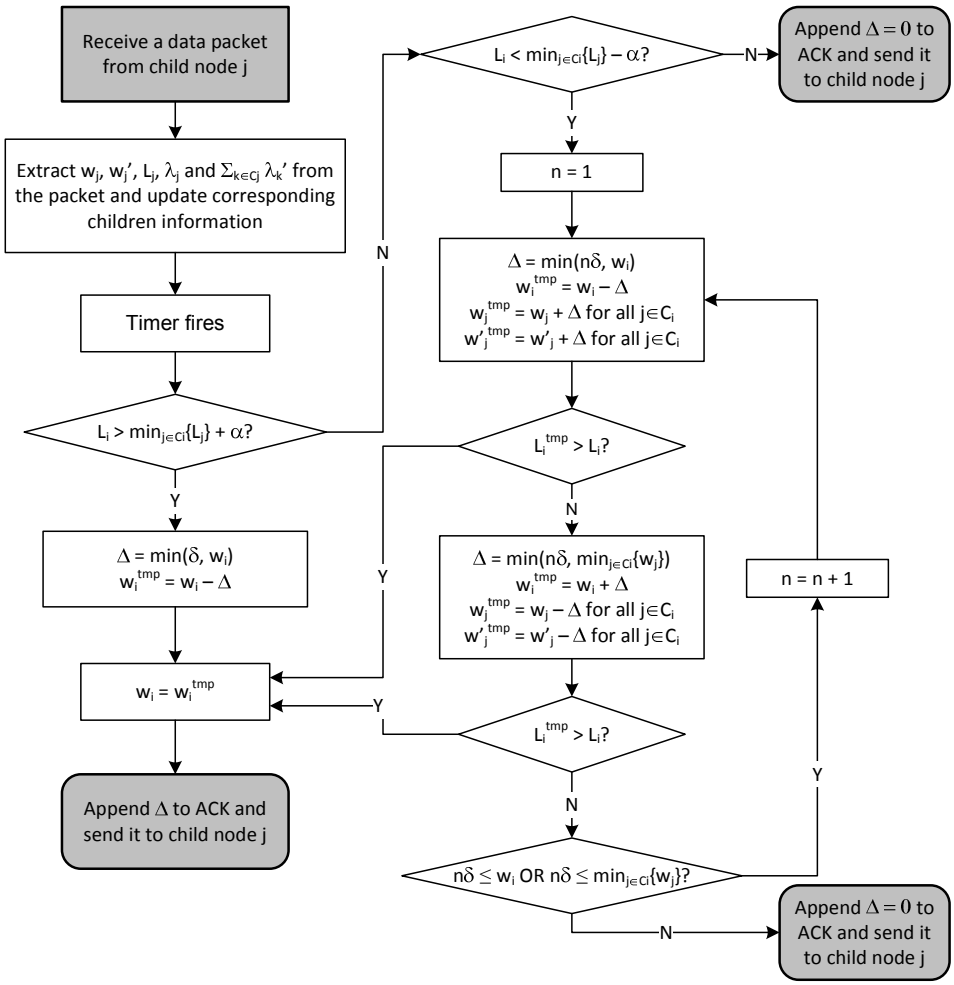


Figure 4.4 Flowchart of adjusting w_i when node i acts as a parent node.

– For every child node j of node i: $w_j = w_j + \Delta$ and $w'_j = w'_j + \Delta$.

The adjustment strategy is based on the following observations. From Equation(4.4), we can see that λ'_j is a non-increasing function of w_j and w'_j. Particularly, λ'_j remains unchanged if $w'_j \le \frac{1}{\lambda_j}$ and $w_j \le \frac{1}{\sum_{k\in C_j}\lambda'_k}$; it increases otherwise. If λ'_j decreases, as node j's input rate does not change, node j's lifetime will be increased according to Equation (4.1). If λ'_j does not change, w_j and w'_j will be further increased in the follow-up adjustment rounds. Hence, at least one of the conditions of $w'_j \le \frac{1}{\lambda_j}$ and $w_j \le \frac{1}{\sum_{k\in C_j}\lambda'_k}$ will be eventually broken, and then node j's lifetime can be extended.

- *Case II*: Node i has shorter lifetime than each child (i.e., $L_i < \min_{j\in C_i} L_j - \alpha$). In this case, the

scheme will attempt to increase the lifetime of node i.

According to Equation (4.1), node i's lifetime is affected by two factors: its data output rate λ_i' and data input rate $\sum_{j \in C_i} \lambda_j'$. To extend lifetime, node i may increase its w_i to reduce its data output rate; on the other hand, its children $j \in C_i$ will have to reduce their w_j values to satisfy the delay requirement. resulting in an increased input rate $\sum_{j \in C_i} \lambda_j'$. The combination of the above two effects may or may not lead to lifetime increase. Similarly, decreasing w_i and increasing w_j for each child i will decrease its data input rate but may increase its data output rate, which may or may not lead to lifetime increase.

Due to the above uncertainty, the scheme will not immediately increase (or decrease) its w_i or ask its children to decrease (or increase) their aggregation delays as in the previous case. Instead, it tries the possible increase/decrease strategies, estimate their effects on its lifetime locally, until an effective strategy has been found or all possible strategies have been tried. If an effective strategy is found, i.e., a Δ (which could be positive or negation) is found such that adding Δ to w_i and subtracting it from w_j of every child j will result in a lifetime increase in node i, the strategy will be applied on node i and Δ will be sent to the children nodes.

4.5.2.2 Behaviors of a Child Node

As shown in Figure 4.5, a child node simply updates its w_i and w_i' according to its parent node's instruction.

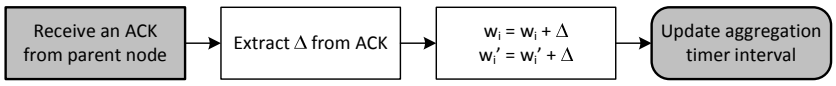


Figure 4.5 Flowchart of adjusting w_i when node i acts as a child node.

4.6 Implementation and Evaluation

4.6.1 Implementation

Our proposed LBA scheme has been implemented in TinyOS 2.1.0 as a middleware component, which takes 248 bytes of RAM and 9180 bytes of ROM for a tree network where each non-leaf node has 10 children nodes on average. This component sits between the application and routing layers. If it is disabled, data from the application layer is sent down to routing layer transparently; otherwise, self-generated data from application layer and received sensory data from routing layer are aggregated inside it, and the aggregated data will be sent to the routing component after the aggregation timer fires. CTP [26] is adopted as the routing layer protocol to build the data collection tree and RI-MAC [50] is used as the MAC layer protocol. RI-MAC is a receiver-initiated protocol for low duty cycled sensor networks, its energy consumption model matches with our lifetime estimation model analyzed in Section 4.4.1.

In the implementation, there are three types of existing messages required by different components and we take advantage of the availability of these messages to enable the aggregation information exchange with the minimum overhead. In each data message, the aggregation information including nodal lifetime, self-data generation interval, aggregation interval, incoming and outgoing intervals are piggybacked, and a parent node can collect these information of its child node when data communication happens between them. In each ACK message (software ACK used in RI-MAC), the updated aggregation interval value is added to notify the child node. In a receiver's beacon (required by RI-MAC), the parent node's lifetime is added.

4.6.2 Testbed Experiment Results

4.6.2.1 Setup

We evaluate the performance of the proposed adaptive assignment scheme (denoted as LBA in figures) in terms of network lifetime and end-to-end data delivery delay. The results are compared to the theoretical upperbound which is computed using Algorithm 4 (denoted as UPPER in figures) and the following two aggregation schemes:

- PTL (pushing aggregation delays to leaves): aggregation delays are only assigned to leaf nodes; the aggregation delays do not change after assignment. The delay assignment is implemented as follows. After tree has been built, the sink broadcasts to all nodes on the tree the maximum allowed aggregation and transmission delay D_p; then, each leaf node i takes $D_p - d_i$ (recalling d_i is its average transmission delay) as its aggregation delay.

- AVG (average assignment of aggregation delays): aggregation delays are assigned to nodes as in the *initial phase* of our proposed scheme; but the aggregation delays do not change after assignment.

We set up a testbed network of 32 TelosB motes, forming a tree topology shown in Figure 4.6, where Node 0 is the sink (i.e., root). RI-MAC parameter T_r and the average data packet generation interval at source nodes are both set to 1 second, RI-MAC parameters ϕ and τ are 40 milliseconds. The end-to-end delivery requirement D changes from 20s to 140s, and $p = 80\%$. At the beginning of each experiment, the initial nodal energy level is uniform or non-uniform. When the initial energy level is uniform, each sensor node's battery is in the full energy level; when it is non-uniform, some nodes start with 1/4, 1/2 or 3/4 of the full energy level, while others still start with the full energy level. To save experiment time, the full energy level is set to 200 Joules which can support a mote running for 45 minutes at 100% radio duty cycle.

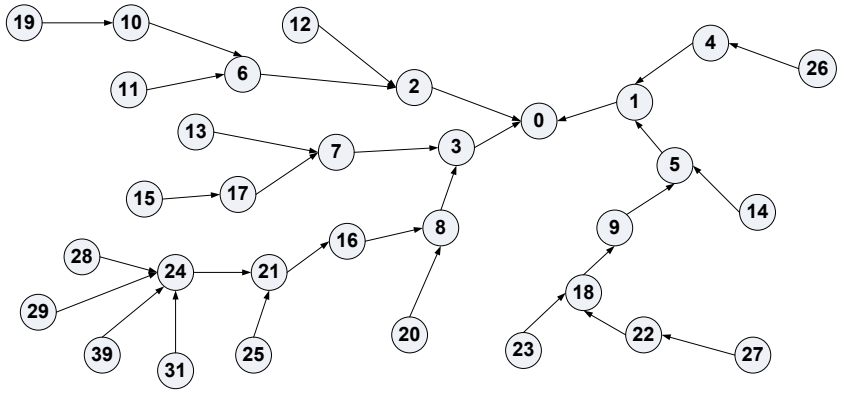


Figure 4.6 Network topology in testbed experiments.

4.6.2.2 Performance Under Uniform Initial Energy Distribution

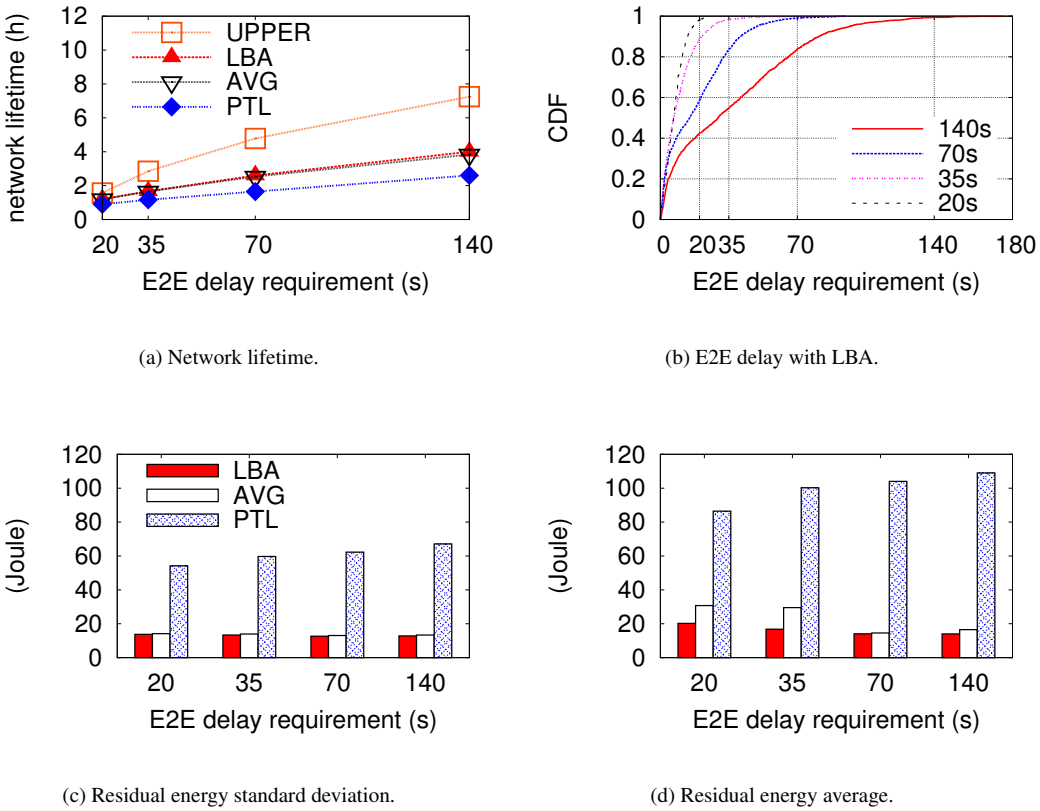
Figures 4.7 and 4.8 show the performance evaluation results when the initial energy levels of nodes are uniform.

When all nodes are data sources As shown in Figure 4.7(a), the network lifetime achieved by LBA and AVG is significantly higher than that of PTL. Note that Figure 4.7(a)(c) and Figure 4.7(a)(d) plot the standard deviation and average of nodal residual energy when the first node dies. This is not surprising because PTL does not allow nodes to aggregate data they forward while LBA and AVG allow nodes to aggregate data generated or forwarded by them. Due to the uniform distribution of nodal energy level, LBA and AVG do not have significant different performance. Indeed, as shown in Figures 4.7(c) and 4.7(d), both schemes result in small deviation in nodal lifetime among all nodes, which indicates that the strategy of fairly and statically assigning aggregation delay adopted by AVG can already achieve good performance and therefore LBA does not bring much extra benefit. Figure 4.7(b) shows the CDF of end-to-end delay when LBA is applied. As can be seen, the end-to-end delay requirements can be met.

When only leaf nodes are data sources As shown in Figure 4.8(a), the network lifetime achieved by LBA and AVG is still significantly higher than that of PTL. The results show that self-aggregation does not fully exploit the aggregation opportunities. Data packets resulted from self-aggregations can be further aggregated at joint points of multiple branches on the tree, and such opportunities can be seized by LBA and AVG. Also due to the uniform distribution of initial nodal energy level, the performance superiority of LBA over AVG is still small. However, as we can see from Figure 4.8(c) and 4.8(d), LBA can balance nodal energy levels more effectively than AVG, which explains the longer lifetime achieved by LBA.

4.6.2.3 Performance Under Non-uniform Initial Energy Distribution

The network lifetime is affected by the degree of deviation in nodal lifetime, which is in turn affected by the degree of deviation in nodal residual energy level. Hence, we also evaluate the performance of aggregation schemes when the initial nodal energy levels are different among nodes. Partic-



(a) Network lifetime.

(b) E2E delay with LBA.

(c) Residual energy standard deviation.

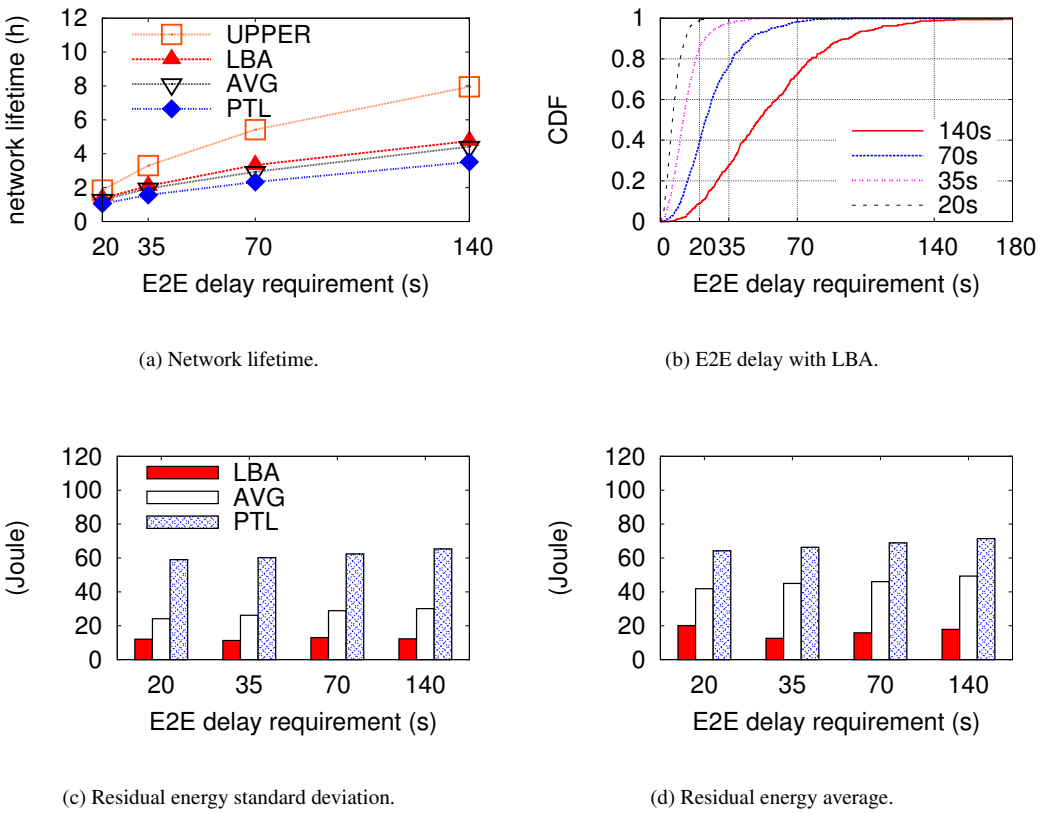
(d) Residual energy average.

Figure 4.7 Performance comparison when all nodes generate data packets.

ularly, in the experiments, we randomly chose a certain percentage of nodes to start working at a lower energy level than others who are with full initial energy level.

Performance comparison Figure 4.9 demonstrates that LBA can achieve larger network lifetime improvement over both AVG and PTL in this scenario. Particularl, two nodes start with a low energy level equal to 1/4 of the full energy level while others start with the full energy level.

As the performance superiority of AVG over PTL has been obvious from previous experiments, we here focus on explaining the difference between LBA and AVG. With LBA and AVG, nodes start with the same distribution of aggregation delays, and the distribution of delays is unaware of the difference in nodal residual energy level or nodal lifetime. With AVG, the distribution of delays does not change after the initial distribution. Hence, if significantly differences of nodal residual energy (and hence lifetime)



(a) Network lifetime.

(b) E2E delay with LBA.

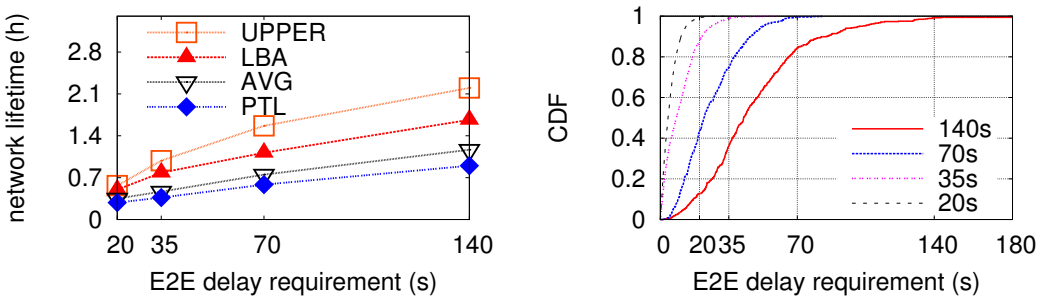
(c) Residual energy standard deviation.

(d) Residual energy average.

Figure 4.8 Performance comparison when only leaf nodes generate data packets.

are present among the nodes, nodes with the lowest nodal lifetime will die much earlier than those with longer nodal lifetime, leading to shortened network lifetime. On the other hand, LBA can dynamically adapt the distribution of aggregation delays among nodes such that nodes with lower nodal lifetime can gain more aggregation delays from those of higher nodal lifetime, leading to balanced distribution of nodal lifetime among nodes and hence prolonged network lifetime.

A working trace To better understand how LBA prolongs the lifetime of low energy (and hence short lifetime) nodes in the network, Figure 4.10 shows the changing trace of aggregation delays and nodal energy of all nodes on the path from node 28 to the sink. In this experiment, node 8 starts with a low energy level equals to 1/4 of the full energy level while others start with the full energy level and $D = 35$ s, $p = 80\%$. We can find that when node 8's own aggregation delay keeps increasing while its



(a) Network lifetime.

(b) E2E delay with LBA.

Figure 4.9 Performance comparison under non-uniform initial nodal energy.

ancestor's (node 3) and successors' (node 16) keep dropping accordingly. When node 16's aggregation delay reaches 0, it gains aggregation delay from its subtree (nodes 21, 24 and 28) to compensate node 8. In other words, node 8 can get help not only from its direct parent or child directly, but also from other nodes in the network indirectly. As a result, node 8's energy drops slowly compared to all other nodes on the path, and the network lifetime is significantly extended.

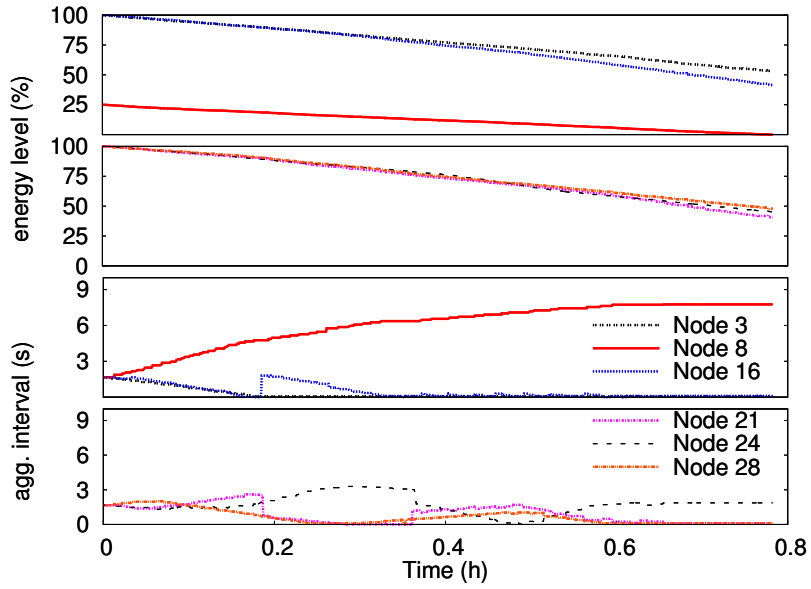
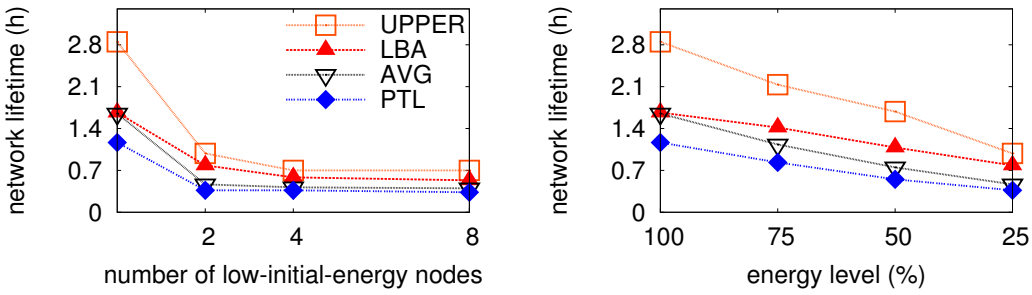


Figure 4.10 Trace of aggregation delays and nodal energy.

Impact of the degree of deviation in nodal energy levels To evaluate the impact of the different degrees of deviation in nodal energy levels, we conduct two sets of experiments.

In the first set, we fix the initial energy level of each low-initial-energy node to 1/4 of the full energy while varying the number of such low-initial-energy nodes in the 32-node network same as the above. As shown in Figure 4.11(a), as the number of low-initial-energy nodes increases, the performance of AVG and PTL does not change much because the lifetime they can achieve can be affected by even a single such low-initial-energy node. The performance of LBA, however, decreases as the number of low-initial-energy nodes increases. This is because, with limited amount of total aggregation delays, less compensation can be obtained by each low-initial-energy node as the number of such nodes increases.

In the second set of experiments, we fix the number of low-initial-energy nodes while varying the initial energy level from 75% to 25% of the full energy and $D = 35$ s, $p = 80\%$. As can be observed from Figure 4.11(b), as the energy level decreases, the performance of AVG and PTL drops much faster than that of LBA. This is because the network lifetime achieved by AVG and PTL is bounded by the shortest-nodal-lifetime node; but with LBA, the nodal lifetime of such nodes can be increased through re-distribution of aggregation delays. Though the decrease of initial energy level demands more re-distribution efforts and thus can decrease the network lifetime, the decrease is shared among the nodes and thus the decreasing speed is slow.



(a) Network lifetime under different number of low-initial-energy nodes.

(b) Network lifetime under different low-initial-energy levels with two fixed low-initial-energy nodes.

Figure 4.11 Impact of the degree of deviation in nodal energy levels.

4.6.2.4 Performance Under Spatial and Temporal Various Data Generation Rates

We evaluate the performance under a more realistic situation where the data generation rates vary during the network operational time. Specifically, in this experiment, each node changes its data generation interval randomly in a specified range after generating every 100 packets and the end-to-end delay requirement D is 35s with $p = 80\%$. Two nodes in the network start working with 1/4 of the full energy while others start with full energy. Figure 4.12 shows the lifetime achieved by three compared schemes as the data generation interval varies in different ranges, where a "1-Xs" label on the X-axis means the range is from 1 to X seconds. As we can see, the performance of LBA is always significantly higher than that of AVG and PTL when the data generation interval range changes. This indicates that LBA can work adaptively to the changes in data generation rates.

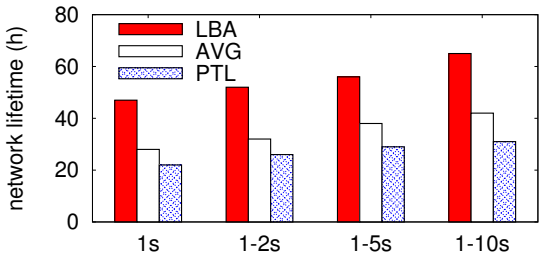


Figure 4.12 Performance comparison under various data generation rates.

4.7 Conclusions

This study introduced a lifetime balanced data aggregation scheme LBA for asynchronous duty cycle sensor networks. Through adaptively adjusting the aggregation holding time between neighboring nodes, LBA can effectively improve the nodal lifetime of nodes of lower energy supplies and/or higher energy consumption and thus prolong the network lifetime, which has been verified by extensive experimental evaluations based on a sensor network testbed.

CHAPTER 5. JOINT AGGREGATION AND MAC DESIGN TO PROLONG SENSOR NETWORK LIFETIME

5.1 Introduction

5.1.1 Motivation

Extending the lifetime of sensor networks is critically important when the networks are deployed for long-term monitoring applications. Besides duty cycling, balancing nodal lifetime is another major approach for network lifetime extension, because the network lifetime is often defined as the minimal nodal lifetime among all nodes in the network [6, 55, 56]. Following this approach, a variety of duty-cycled MAC protocols [68–70] and data aggregation schemes [54, 71] have been proposed recently.

Duty-cycled sensor networks rely on MAC protocols to establish rendezvous between sender and receiver nodes. The incurred MAC-layer communication overhead is distributed between sender and receiver in different manners with different MAC protocols. To balance nodal lifetime, the authors of [68–70] proposed to adapt the distribution of communication overhead among neighbors according to their nodal lifetime, i.e., longer-lifetime nodes shall absorb more communication overhead than their shorter-lifetime neighbors.

Through eliminating inherent redundancy in raw sensory data, in-network data aggregation [52,53] can effectively reduce network traffic. For the sake of aggregation, a node needs to hold data (received or self-generated) for a while. Clearly, a node can suppress more data traffic with a longer holding time. However, holding data introduces extra delay, and the value of sensory data could be greatly depreciated if the data is delivered to the sink with a delay longer than an application-specified delay bound. To balance nodal lifetime while maintaining a required end-to-end delay bound, the authors of [54, 71] proposed to adapt the distribution of holding time among nodes along the same route according to their

nodal lifetime; i.e., shorter-lifetime nodes shall hold data longer to reduce its outgoing data traffic while longer-lifetime nodes shall hold data for a shorter time so that the end-to-end delay requirement is still guaranteed.

As the aforementioned MAC and data aggregation schemes improve the network lifetime from two distinct perspectives, it is naturally an attractive idea to explore the possibility of a joint design of MAC and data aggregation to further improve the network lifetime. This, however, is a challenging task due to the following reasons. Firstly, most of the MAC protocols were designed without explicitly considering the end-to-end delay requirement. As a result, they may yield uncontrollable end-to-end delay under certain practical scenarios; this is unacceptable for data aggregation applications that often require a stringent end-to-end delay bound. Secondly, even if there exists a MAC protocol that can provide a certain delay guarantee, when it works with a data aggregation scheme, it is non-trivial to decide how to divide the allowed end-to-end delay into two parts to serve as the delay constraints respectively for the MAC and data aggregation protocols, such that the maximum lifetime improvement can be accomplished. Therefore, it is important to develop an innovative approach to integrate MAC and data aggregation protocols together via a joint adjustment of their protocol parameters.

5.1.2 Literature Survey

Research has been conducted on designing duty-cycled MAC protocols and energy-efficient or lifetime-extending data aggregation schemes. However, there is no research on jointly adjusting data aggregation and MAC behaviors to extend the sensor network lifetime.

Numerous MAC protocols [68–70, 72] have been proposed to extend network lifetime through balancing nodal lifetime. Particularly, SEESAW [68] balances the energy consumption between sender and receiver through adapting the data retry interval at the sender side and the channel checking period at the receiver side. ZeroCal [70] targets at improving the fairness of energy utilization in duty-cycled sensor networks by dynamically tuning the nodal wakeup interval. GDSIC [69] decides the individual nodal wakeup interval through solving distributed convex optimization problems. Though the network lifetime can be prolonged by these schemes, they do not consider the end-to-end delay bound. pTunes [72] is a recently-proposed centralized solution, which formulates a multi-objective optimiza-

tion problem, where prolonging network lifetime and guaranteeing the end-to-end delay can be solved together.

Most of existing data aggregation schemes [60, 62, 65, 73, 74] have the objective of minimizing the total network energy consumption instead of extending network lifetime, or do not consider the end-to-end delay requirement. The problem of balancing nodal lifetime under a delay constraint has been studied in [54, 71]. Particularly, Becchetti et al. [54] investigated the problem of energy-efficient data aggregation within a delay bound, and proposed two distributed schemes to balance energy consumption among sensor nodes. LBA [71] is a recently-proposed lifetime-balancing aggregation protocol. Through dynamically adjusting the aggregation holding time among neighbors to balance their nodal lifetime, LBA provides a low cost, asynchronous, and delay-constrained data aggregation scheme for duty-cycled sensor networks.

5.1.3 Contributions

In this study, we propose a novel holistic approach, called JAM (Joint Aggregation and MAC), to jointly adjust MAC and data aggregation behaviors to extend the sensor network lifetime. The key idea of JAM is to coordinate the aggregation and MAC behaviors at each individual node as well as between neighbors, with the target of extending the minimal nodal lifetime in the neighborhood. As such coordination occurs in all neighborhoods, the network lifetime, i.e., the minimal nodal lifetime among all nodes in the network, may be improved. As JAM reduces both network traffic (via data aggregation) and communication overhead (via duty-cycled MAC), it prolongs the network lifetime more efficiently and effectively than previous works that only use one of the two techniques. The contributions of this work are summarized as follows.

- To the best of our knowledge, JAM is the first design on integrating and jointly configuring MAC and data aggregation protocols to extend the lifetime of duty-cycled sensor networks.
- JAM is a distributed and lightweight solution. It works through limited control information exchanged locally between neighbors.
- JAM has been implemented and evaluated on a sensor network testbed, and results show that it

can achieve significant improvement on network lifetime compared to the state-of-the-art solutions.

5.2 System Model and Problem Statement

5.2.1 System Models

We consider a sensor network deployed for monitoring applications where in-network data aggregation is allowed. Each sensor node generates and reports sensory data periodically, and all nodes form a data collection tree rooted at the sink. Protocols like CTP (Collection Tree Protocol) [26] could be used to build and maintain the data collection tree.

5.2.1.1 Aggregation Model

The *total aggregation* model [66] is adopted, which allows an arbitrary number of data packets generated and/or received at the same node to be suppressed into a single data packet. Such a model is useful in many sensor network applications, for example, when users are more interested in the maximum, minimum, average, or percentile statistics of sensory data, rather than the raw data themselves.

With this model, a source node may not send out a sensory data packet immediately after it is generated. Instead, the node may wait for a certain period of time (called the *self-aggregation delay* (SAD)), and aggregates all data generated during the period to reduce the amount of data traffic to its parent node. Similarly, a forwarding node may not forward a data packet immediately after reception; it may wait for another period of time (called the *forwarding-aggregation delay* (FAD)) and aggregate all packets received during the period. Generally, the longer time a node waits for aggregation, the more data traffic can be suppressed; at the same time, data delivery latency is increased.

5.2.1.2 MAC Model

To conserve energy without time synchronization overhead, it is desired to employ an asynchronous and duty-cycled MAC protocol for long-term monitoring applications. The design principle of our proposed scheme does not require a particular MAC protocol. Instead, our design works compatibly with any asynchronous duty-cycled MAC protocol, as long as the protocol allows a node's duty cycle

to be adjusted dynamically. To simplify the presentation, however, we assume each node runs an RI-MAC [50] like protocol as shown in Figure 5.1.

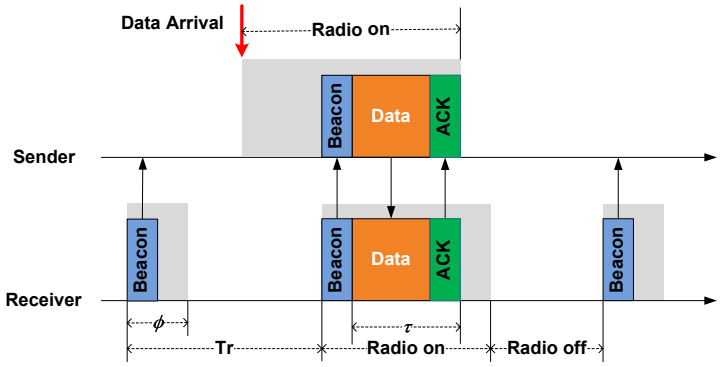


Figure 5.1 An RI-MAC like protocol but with a tunable Tr parameter.

In RI-MAC, each node wakes up every Tr interval to interact with potential senders. Upon wakeup, it sends out a beacon and then checks the channel activity for ϕ time. If a data packet is received within ϕ, it replies with an ACK; otherwise, it goes back to sleep. On the other hand, if a node has a data packet to send, it remains awake and waits idly for the receiver's beacon to start data transmission. In the worst-case scenario, the sender has to stay awake for Tr time before rendezvous with the receiver, which incurs a transmission delay of Tr. Different from the original RI-MAC protocol that has a fixed Tr, we assume Tr is dynamically tunable. As can be observed, a larger Tr reduces the receiver's channel polling frequency and hence its energy consumption; on the other hand, it increases the sender's energy consumption on idly waiting for the beacon, and meanwhile increases the transmission delay over the sender-to-receiver link.

5.2.1.3 Delay Model

Three types of delays are involved along a source-to-sink path: (i) one SAD at the source node, (ii) multiple FADs at the forwarding nodes, and (iii) multiple transmission delays over the links. With the aggregation and MAC models described above, to guarantee that the maximum end-to-end packet delivery delay from any source node i to the sink node is bounded by an application-specific parameter

D, we need to ensure that[1]

$$\text{SAD}(i) + \sum_{m \in \mathcal{S}_i} (\text{Tr}(m) + \text{FAD}(m)) \leqslant D, \tag{5.1}$$

where $\mathcal{S}_i$ is the path from node i's parent to sink. As larger aggregation delays would allow more traffic to be aggregated to save more energy, the delay bound shall be fully utilized to maximize network lifetime, meaning that the equality shall hold true in Inequality (5.1) in practical schemes.

5.2.2 Problem Statement

To effectively prolong the sensor network lifetime under the end-to-end packet delivery delay constraint, it is critical to have a holistic approach to adjust data aggregation and MAC behaviors of all sensor nodes. Ideally, all sensor nodes shall work together to maximize the minimum nodal lifetime in the entire network. Unfortunately, this global objective is impossible to accomplish in a realistic sensor network, as it requires each node to know the residual energy levels and data generation rates of all other nodes, and the topology of the network, which are highly dynamic and often unpredictable by nature. Instead, we study the following localized problem for each sensor node i in the network:

Objective:

- $\max \min_{j \in \{i\} \cup \mathcal{C}(i)} L(j)$, where $L(j)$ is j's nodal lifetime and its computation will be explained later in Section 5.3.3. $\mathcal{C}(i)$ is the set of i's child nodes.

Subject to:

- $\text{SAD}(i),\ \text{FAD}(i),\ \text{Tr}(i) \geqslant 0$;
- *End-to-End Delay Requirement:*

$$\text{SAD}(i) + \sum_{m \in \mathcal{S}_i} (\text{Tr}(m) + \text{FAD}(m)) = D. \tag{5.2}$$

Output:

[1] In practice, data or ACK packets may get lost due to collision, interference, or deteriorated channel condition. As a result, the sensor node may need to retransmit multiple times before the data packet can be delivered successfully. This issue has been dealt with in JAM (as shown in Section 5.3.6.2) by replacing Tr with ETX $\cdot$ Tr in Equations (5.1) and (5.2), where ETX is the expected number of transmission attempts to deliver a data packet successfully over one hop. To simplify the presentation, we set ETX $= 1$ during the explanation of the JAM design in the next section, while the practical Equation (5.14) (given in Section 5.3.6.2) is used in the actual JAM implementation.

- i's MAC protocol parameter: Tr(i);
- i's data aggregation parameters: SAD(i) and FAD(i).

The goal of this problem is to maximize the minimal nodal lifetime in i's neighborhood. As such procedure occurs in all neighborhoods, the minimal nodal lifetime in the entire network, i.e., the network lifetime, may be improved gradually.

5.3 The Proposed JAM Scheme

In this section, we propose a protocol called *JAM* to address the problem defined above. In JAM, coordination only occurs between a sensor node and its child nodes, through exchanging lightweight control information as well as adjusting their aggregation and MAC behaviors together in a collaborative manner. Figure 5.2 gives an overview of the JAM scheme. To ease the presentation, we use the topology shown in Figure 5.3 as an example to explain the design details of JAM.

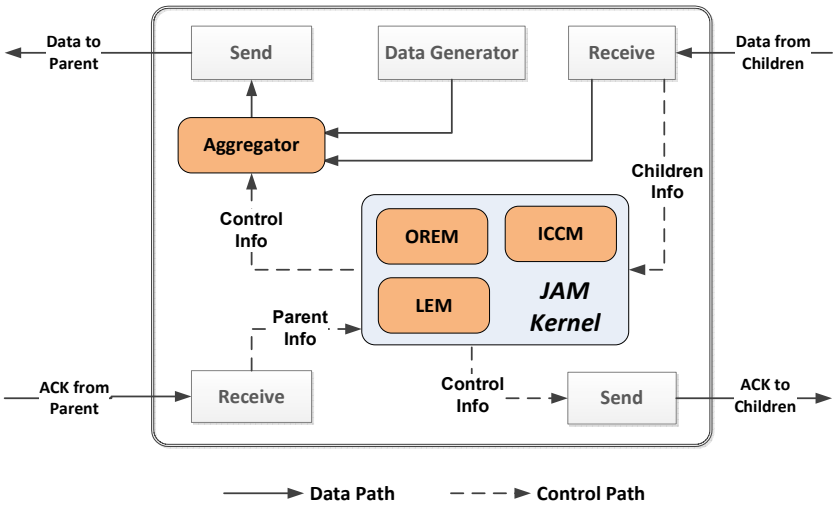


Figure 5.2 JAM Overview.

JAM consists of four modules: Aggregator, OREM (Output Rate Estimation Module), LEM (Lifetime Estimation Module), and ICCM (Intra-node Cross-layer Collaboration Module). In general, when node i receives a data packet from its child nodes or an ACK from its parent, it extracts the control information embedded in the packet and feeds them into JAM kernel. Here, the control information

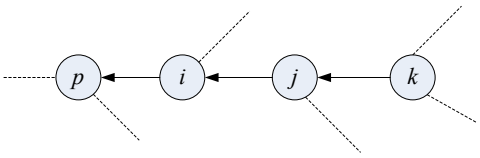


Figure 5.3 Topology used to describe JAM details.

needed by JAM kernel includes five items from each child node j of i: $e(j)$, $\lambda(j)$, SAD(j), the delay introduced at node j: $D(j) = \text{FAD}(j) + \text{Tr}(j)$, and j's own input data rate $\sum_{k \in \mathcal{C}(j)} \mu(k)$ where $\mu(k)$ is the output data rate of a child node k of j; and two items from parent node p of i: Tr(p) and $\theta(p)$. We will explain the meaning of θ in Section 5.3.5. With these information, JAM kernel decides how node i shall adjust its MAC behavior (i.e., Tr) and aggregation behavior (i.e., SAD and FAD) to improve the minimal nodal lifetime in its neighborhood. The decision is then piggybacked into the ACK packet to its child nodes. Upon being notified, each child node adjusts its MAC and aggregation behaviors.

5.3.1 Aggregator Module

The aggregator module controls the nodal aggregation behavior. Specifically, it works as follows at node i:

Case I: $\frac{1}{\sum_{j \in \mathcal{C}(i)} \mu(j)} \leqslant$ *FAD*$(i) \leqslant$ *SAD*(i). In this case, node i has a data packet arrival rate no lower than $1/\text{FAD}(i)$ and $1/\text{SAD}(i)$. A timer is fired every FAD(i) interval. When it fires, all data packets received and self-generated since its last firing are aggregated into a single packet, and then forwarded to the parent node. This way, we ensure that packets received from i's child nodes are held for no longer than FAD(i) time. Also, since the self-generated data packets are aggregated and forwarded every FAD(i) time, which is always smaller than SAD(i) in JAM, their delay requirement is guaranteed as well. We will prove FAD$(i) \leqslant$ SAD(i) for any node i in Section 5.3.2.

Case II: $\frac{1}{\sum_{j \in \mathcal{C}(i)} \mu(j)} >$ *FAD*(i) *and* $\frac{1}{\lambda(i)} \leqslant$ *SAD*(i). In this case, data packets from child nodes arrive at a rate lower than $1/\text{FAD}(i)$ while the node itself generates data packets at a rate no lower than $1/\text{SAD}(i)$. Data received from children and self-generated data are treated differently as follows:

- Whenever a data packet is received from a child node, the packet is aggregated immediately with all the self-generated data packets that have not yet been aggregated, and then forwarded to the

parent node. Hence, the forwarding-aggregation delay is zero.

- A timer is fired every $\text{SAD}(i)$ interval. When it fires, all the self-generated data packets that have not yet been aggregated are suppressed into a single packet and then forwarded to the parent node.

Case III: $\frac{1}{\sum_{j\in\mathcal{C}(i)}\mu(j)} > \textit{FAD}(i)$ *and* $\frac{1}{\lambda(i)} > \textit{SAD}(i)$. In this case, data packets from child nodes arrive at a rate lower than $1/\text{FAD}(i)$ while the node itself generates data packets at a rate lower than $1/\text{SAD}(i)$. As the data packet arrival/generation rates are low, every data packet is simply forwarded to the parent node immediately upon its reception or generation.

5.3.2 Output Rate Estimation Module (OREM)

This model is part of the JAM kernel (shown in Figure 5.4) and is used to estimate the nodal output data rate. Before explaining how it works in detail, we first prove $\text{FAD}(i) \leqslant \text{SAD}(i)$ for any node i. According to the end-to-end delay requirement in Equation (5.2), node i shall satisfy:

$$\text{SAD}(i) + \sum_{m\in\mathcal{S}_i} (\text{Tr}(m) + \text{FAD}(m)) = D.$$

Similarly, for every child node j of i, we have:

$$\text{SAD}(j) + (\text{FAD}(i) + \text{Tr}(i)) + \sum_{m\in\mathcal{S}_i} (\text{Tr}(m) + \text{FAD}(m)) = D.$$

Hence, it follows that:

$$\text{FAD}(i) \leqslant D - \sum_{m\in\mathcal{S}_i} (\text{Tr}(m) + \text{FAD}(m)) = \text{SAD}(i). \tag{5.3}$$

With the four inputs: $\text{FAD}(i)$, $\text{SAD}(i)$, $\sum_{j\in\mathcal{C}(i)}\mu(j)$, and $\lambda(i)$, OREM estimates the output data rate of node i according to the three cases described in Section 5.3.1 as follows:

$$\mu(i) = \begin{cases} \frac{1}{\text{FAD}(i)} & : \frac{1}{\sum_{j\in\mathcal{C}(i)}\mu(j)} \leqslant \text{FAD}(i), \\ \sum_{j\in\mathcal{C}(i)}\mu(j) + \frac{1}{\text{SAD}(i)} & : \frac{1}{\sum_{j\in\mathcal{C}(i)}\mu(j)} > \text{FAD}(i) \\ & \text{and } \frac{1}{\lambda(i)} \leqslant \text{SAD}(i), \\ \sum_{j\in\mathcal{C}(i)}\mu(j) + \lambda(i) & : \frac{1}{\sum_{j\in\mathcal{C}(i)}\mu(j)} > \text{FAD}(i) \\ & \text{and } \frac{1}{\lambda(i)} > \text{SAD}(i). \end{cases} \tag{5.4}$$

Note that a node's input data rate is simply the sum of the output data rates from all of its children.

5.3.3 Lifetime Estimation Module (LEM)

The LEM module is another module in the JAM kernel and is used to estimate the nodal lifetime. Its input consists of $e(i)$, Tr(i), Tr(p) (i.e., Tr of node i's parent node p), $\mu(i)$, and $\sum_{j\in\mathcal{C}(i)}\mu(j)$. Its output is the estimated nodal lifetime $L(i)$, which is computed as follows:

$$L(i) = \frac{e(i)}{c(i)}, \tag{5.5}$$

where $e(i)$ is the residual energy and $c(i)$ is the energy consumption rate:

$$c(i) = \left(\frac{\text{Tr}(p)}{2} + \tau\right)\mu(i)P + \frac{\phi}{\text{Tr}(i)}P + \tau\sum_{j\in C(i)}\mu(j)P. \tag{5.6}$$

Here, P is the power consumption rate when a node's radio is on. To send a data packet, node i waits for $\frac{\text{Tr}(p)}{2}$ time on average for its parent node p to wake up and then spends τ time for the transmission. Hence, it consumes $\left(\frac{\text{Tr}(p)}{2} + \tau\right)\mu(i)P$ power on average for data transmissions. The second term in Equation (5.6) represents the average amount of power consumed to monitor channel for ϕ time every Tr(i) interval, while the third term is the average amount of power consumed for data receptions. As radio is the most energy-consuming component, we ignore other energy consumptions such as sensing and computation, which could be easily plugged into Equation (5.6).

5.3.4 Intra-node Cross-layer Collaboration Module (ICCM)

According to Equation (5.2), SAD value at the source node can be computed once the FAD and Tr values of its ancestor nodes have been determined. As both MAC and aggregation behaviors affect nodal lifetime, their behaviors should be coordinated. Specifically, as each forwarding node i introduces a delay of $D(i) = \text{FAD}(i) + \text{Tr}(i)$, the objective of the ICCM module in the JAM kernel is to determine proper FAD(i) and Tr(i) values for a given $D(i)$, so that nodal lifetime $L(i)$ is maximized (denote as $L^*(i)$). To achieve this goal, there are two cases to consider:

Case I: FAD$(i) < \dfrac{1}{\sum_{j\in\mathcal{C}(i)}\mu(j)}$. Depending on the relation between SAD(i) and $\lambda(i)$, $\mu(i)$ equates the second or third case of Equation (5.4). In these two cases, $\mu(i)$ is not affected by FAD(i) and hence nodal lifetime is only affected by Tr(i) according to Equations (5.5) and (5.6). As the energy consumption rate ($c(i)$) in Equation (5.6) is a decreasing function of Tr(i), optimal nodal lifetime is

achieved when $D(i)$ is fully allocated to $\text{Tr}(i)$ and hence $\text{FAD}(i) = 0$:

$$L^*\left(i \;\middle|\; \text{FAD}(i) < \frac{1}{\sum_{j\in\mathcal{C}(i)}\mu(j)}\right) = L(i \mid \text{FAD}(i) = 0). \tag{5.7}$$

Case II: FAD$(i) \geqslant \frac{1}{\sum_{j\in\mathcal{C}(i)}\mu(j)}$. In this case, the output data rate is $\mu(i) = \frac{1}{\text{FAD}(i)}$ according to Equation (5.4). In order to achieve optimal nodal lifetime in this case, we need to minimize the following term according to Equation (5.6):

$$\left(\frac{\text{Tr}(p)}{2} + \tau\right)\frac{1}{\text{FAD}(i)} + \frac{\phi}{\text{Tr}(i)}, \tag{5.8}$$

which we denote as $f(\text{FAD}(i))$. To ease the presentation, we use α to denote $\frac{\text{Tr}(p)}{2} + \tau$. Plugging in $\text{Tr}(i) = D(i) - \text{FAD}(i)$, we can rewrite $f(\text{FAD}(i))$ as:

$$f(\text{FAD}(i)) = \frac{\alpha}{\text{FAD}(i)} + \frac{\phi}{D(i) - \text{FAD}(i)}. \tag{5.9}$$

By solving $f(\text{FAD}(i))' = 0$, we have:

$$L^*\left(i \;\middle|\; \text{FAD}(i) \geqslant \frac{1}{\sum_{j\in\mathcal{C}(i)}\mu(j)}\right) =$$
$$\begin{cases} L\left(i \;\middle|\; \text{FAD}(i) = \frac{D(i)}{1+\sqrt{\frac{\phi}{\alpha}}}\right) : \text{when } \frac{1}{\sum_{j\in\mathcal{C}(i)}\mu(j)} \leqslant \frac{D(i)}{1+\sqrt{\frac{\phi}{\alpha}}} \leqslant D(i), \\ \max\left\{L\left(i \mid \text{FAD}(i) = D(i)\right), L\left(i \;\middle|\; \text{FAD}(i) = \frac{1}{\sum_{j\in\mathcal{C}(i)}\mu(j)}\right)\right\} \\ \qquad\qquad : \text{otherwise.} \end{cases} \tag{5.10}$$

To summarize, for a given $D(i)$, $L^*(i)$ and the corresponding optimal $\text{FAD}(i)$ and $\text{Tr}(i)$ can be computed by:

$$L^*(i) = \max\{\text{Equation } (5.7), \text{Equation } (5.10)\}, \tag{5.11}$$

$$\text{FAD}^*(i) = \arg_{\text{FAD}(i)}\max\{L^*(i)\}, \tag{5.12}$$

$$\text{Tr}^*(i) = D(i) - \text{FAD}^*(i). \tag{5.13}$$

5.3.5 JAM Kernel

As the core of the JAM scheme, the JAM kernel of node i is triggered to execute periodically every W time, or on demand whenever its parent node p changes $\text{Tr}(p)$ or $\text{FAD}(p)$ and consequently the delay $D(p)$. We use $\theta(p)$ to denote the change in $D(p)$. The JAM kernel is executed to decide:

- how the extra delay $\theta(p)$ introduced at parent node p shall be absorbed by node i (in the amount of $\theta(p) - \Delta$ via updating $D(i)$ to $D'(i) = D(i) - (\theta(p) - \Delta)$) and its child nodes (in the amount of Δ via updating $D(j)$ to $D'(j) = D(j) - \Delta$); and
- how node i shall split $D'(i)$ into $\text{Tr}'(i)$ and $\text{FAD}'(i)$,

so that the minimal nodal lifetime within node i's neighborhood can be increased. Detailed working procedure of the JAM kernel is illustrated in Figure 5.4 and explained below, while the effect of W will be discussed in Section 5.3.6.3. The figure only shows the interactions between ICCM, OREM, and LEM modules for child node j of i. In the actual JAM implementation, these interactions are executed for every child node of i.

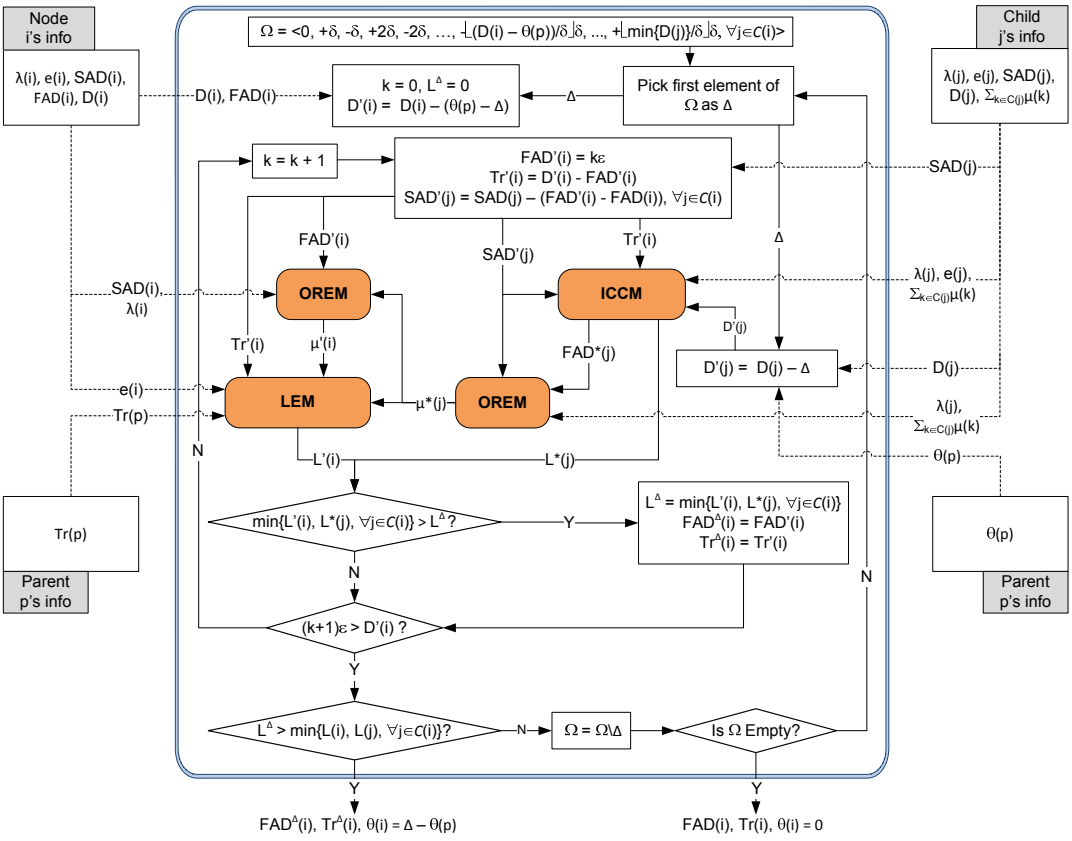


Figure 5.4 JAM kernel of node i.

The JAM kernel of node i takes the following inputs: $\text{Tr}(p)$ and $\theta(p)$ from parent node p, and $\lambda(j)$, $\text{SAD}(j)$, $D(j)$, and $\sum_{k \in C(j)} \mu(k)$ from each child node j. Based on these inputs, it checks the

candidate values of Δ from an ordered sequence $\Omega = \langle 0, \delta, -\delta, 2\delta, -2\delta, \cdots \rangle$ one by one till the first feasible Δ value has been found to increase the minimal nodal lifetime within node i's neighborhood, or till all values in Ω have been exhausted. Here, δ is a system parameter. To ensure $D'(i) \geqslant 0$, the smallest element in Ω is set to $-\left\lfloor \frac{D(i)-\theta(p)}{\delta} \right\rfloor \delta$. Similarly, to ensure $D'(j) \geqslant 0$, the largest element in Ω is set to $\left\lfloor \frac{\min_{j \in C(i)} D(j)}{\delta} \right\rfloor \delta$. For each candidate value of Δ, the JAM kernel executes the following:

- Iteratively, FAD$'(i)$ takes a value from range $[0, D'(i)]$ with a small step ε. Here, ε is a system parameter. For each FAD$'(i)$ value, Tr$'(i) = D'(i) -$ FAD$'(i)$.

- Based on FAD$'(i)$, Tr$'(i)$, and the inputs from each child node j, the ICCM module is called to compute the maximum lifetime that node j can achieve, denoted as $L^*(j)$, and the corresponding FAD$^*(j)$, according to Equations (5.11) and (5.12), respectively.

- FAD$^*(j)$ is fed into the OREM module to compute the output data rate $\mu^*(j)$ of node j. With $\mu^*(j)$ from all child nodes, the output data rate $\mu'(i)$ of node i is also computed using the OREM module.

- Then, the LEM module is called to estimate the nodal lifetime $L'(i)$. If $\min\{L'(i), L^*(j), \forall j \in \mathcal{C}_i\}$ is larger than the highest L^Δ that has been found so far, L^Δ is updated, and the corresponding FAD$'(i)$ and Tr$'(i)$ values are recorded as FAD$^\Delta(i)$ and Tr$^\Delta(i)$.

- When FAD$'(i)$ reaches the boundary condition, i.e., FAD$'(i) > D'(i)$, if the best achievable L^Δ improves the minimal nodal lifetime within node i's neighborhood, i.e., $L^\Delta > \min\{L(i), L(j), \forall j \in \mathcal{C}_i\}$, FAD$^\Delta(i)$ and Tr$^\Delta(i)$ are output, $\theta(i) = \Delta - \theta(p)$ is appended to ACK packets to all child nodes, and search completes; otherwise, the next candidate Δ value will be tested.

Note that, if all the candidate values of Δ in Ω have been checked but none of them improves the minimal nodal lifetime within node i's neighborhood, FAD(i) and Tr(i) remain unchanged and $\theta(i) = 0$.

In our implementation of the JAM scheme, we set $\delta = 0.5$ s and $\varepsilon = 0.1$ s. This means that, when the average per-hop delay is 10 s, there is a total of 40 candidate values for Δ, and 100 candidate values for FAD$'$. Therefore, in the worst-case scenario, the JAM kernel needs to iterate 4000 times to

complete the execution, which is acceptable in practice. To further reduce the complexity, we adopt a two-step heuristic as follows. We first use a larger ε value (i.e., 1 s) to conduct the initial search; when a feasible FAD$'$ has been found, we use a smaller ε value (i.e., 0.1 s) to refine the search around it. With this simple heuristic, the search space for FAD$'$ is reduced to 30 and the total number of iterations in the worst-case scenario is reduced to 1200 in the above example.

5.3.6 Other Design Considerations

5.3.6.1 JAM Initialization

After the data collection tree has been established (i.e., the routing table of each node becomes stable), each node i needs to decide its initial FAD(i) and Tr(i) values. In JAM, all nodes start with a default Tr value and the rest of the end-to-end delay bound is evenly distributed to nodes along the source-to-sink path. For this purpose, each node i periodically measures the most updated (i) accumulative delays from its parent node to the sink ($D(i \rightarrow s)$), i.e., the second term in the end-to-end delay requirement in Equation (5.2), and (ii) hop count to its farthest descendant ($h(i)$).

We assume that the sink's radio is always on and its wakeup interval is Tr(s) = 0 s. As the sink is the end of data delivering paths, FAD(s) = 0 s. $D(s \rightarrow s) = 0$ s is then broadcast to all of its children. Upon receiving $D(i \rightarrow s)$, each child node j acts as follows:

- If node j is not a leaf node, it sets $D(j) = \frac{D - D(i \rightarrow s)}{h(j)+1}$, FAD($j$) = $D(j)$ − Tr(j), SAD(j) = $D - D(i \rightarrow s)$, and sends $D(j \rightarrow s) = D(i \rightarrow s) + D(j)$ to all of its children. In the example shown in Figure 5.5, node 1 sets $D(1) = 15/3 = 5$ s, FAD(1) = 4 s, SAD(1) = 15 s, and sends $D(1 \rightarrow s) = 5$ s to nodes 2 and 3.

- If node j is a leaf node, it simply sets SAD(j) = $D - D(i \rightarrow s)$. In the example shown in Figure 5.5, leaf node 4 sets SAD(4) = 15 − 10 = 5 s.

5.3.6.2 Handling of Packet Loss

In practice, data or ACK packets may get lost due to collision, interference, or deteriorated channel condition, and the sensor node may need to retransmit multiple times before the data packet can be

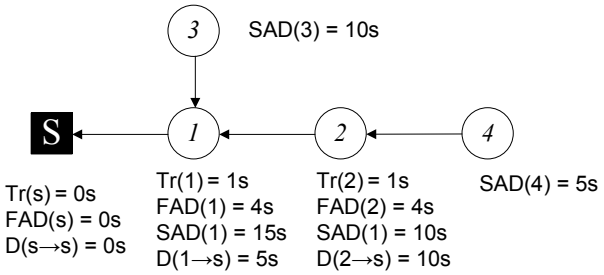


Figure 5.5 JAM initialization example where the default Tr is 1 s and D is 15 s.

delivered successfully. This issue has been dealt with in JAM by replacing Tr with ETX $\cdot$ Tr in the end-to-end delay requirement (i.e., Equation (5.2) in Section 5.2.2), where ETX is the expected number of transmission attempts to deliver a data packet successfully over one hop:

$$\text{SAD}(i) + \sum_{m \in \mathcal{S}_i} (\text{ETX}(m^c, m)\text{Tr}(m) + \text{FAD}(m)) = D. \tag{5.14}$$

Here, m^c is the child node of m along the path from i to the sink. Similarly, Equations (5.6) and (5.10) have also been updated to include the ETX information. Note that measurement of ETX is readily available in many routing protocols such as CTP [26], and thus not an extra overhead.

5.3.6.3 Adaptive Adjustment Interval W

The JAM kernel is triggered to execute periodically every W interval, which poses a tradeoff. A small interval, i.e., frequent adjustment, makes JAM more responsive to the changes in the network and allows it to approach a balanced nodal lifetime distribution in the network sooner, but uses more computational resources. A large interval, on the other hand, uses less computational resources but may let nodes stay in suboptimal states for a longer time.

We adopt an adaptive approach to adjust W dynamically (between $W_{\min}$ and $W_{\max}$) to the network condition. Specifically, if the current state is already the best that can be found, i.e., no Δ in Ω improves the minimal nodal lifetime within the neighborhood as described in Section 5.3.5, W is doubled till reaching $W_{\max}$. Otherwise, W is reset to $W_{\min}$. This way, when the nodal lifetime distribution is heterogeneous or the network configuration changes, JAM allows the network to re-converge quickly to the balanced nodal lifetime distribution; otherwise, it decreases the adjustment frequency exponentially

to save the control overhead in the long term. In the JAM implementation, we set $W_{\min}$ and $W_{\max}$ conservatively to 1 minute and 16 minutes, respectively.

5.4 JAM Implementation

5.4.0.4 Software Component

We have implemented JAM in TinyOS 2.1.0. As shown in Figure 5.6, the shaded parts illustrate the core components of JAM in the software architecture: (i) the aggregation component that sits between application and routing layers, and (ii) the MAC component that is designed based on RI-MAC [50]. CTP [26] is adopted as the routing layer protocol to set up the data collection tree.

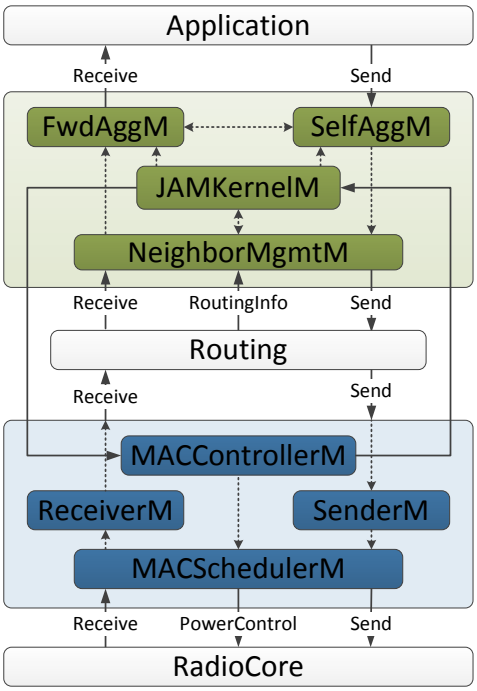


Figure 5.6 Implementation of JAM in TinyOS.

When data packets from a child node j arrives at node i's MAC layer, the piggybacked control information will be extracted, passed to, and processed at the *JAMKernelM* module, which implements the JAM kernel. The *NeighborMgmtM* module manages all senders' information while the *FwdAggM* and *SelfAggM* modules maintain the aggregation timers. After deciding the new Tr(i) and FAD(i) values, the JAMKernelM module notifies the *MACControllerM* module to adopt the latest wakeup

interval, and the FwdAggM and SelfAggM modules to adjust the aggregation timers. JAMKernelM also informs MACControllerM of $\theta(i)$ and piggybacks it in the ACKs to child nodes.

5.4.0.5 Hardware Component

Among all the control information piggybacked in data packets, nodal residual energy is an important piece. As TelosB motes do not provide an interface to measure nodal residual energy, we have designed and fabricated a TelosB power meter kit as shown in Figure 5.7. This kit measures the nodal power consumption rate, based on which we can calculate the total energy consumed so far. The nodal residual energy is then the difference between the battery energy capacity [75] and the consumed energy.

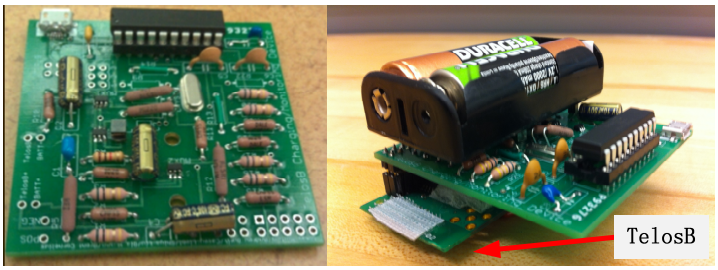


Figure 5.7 TelosB power meter kit used in JAM.

5.5 Performance Evaluation

NS-2 based simulations and TinyOS testbed experiments have been conducted to evaluate the JAM performance in terms of *network lifetime*, *average nodal power consumption*, and *end-to-end delivery delay*. We compare JAM with a naive scheme called AVG which simply sets FAD and Tr values according to Section 5.3.6.1 without runtime adjustment, and LBA [71] which is a state-of-the-art lifetime-balancing aggregation protocol under delay constraint.

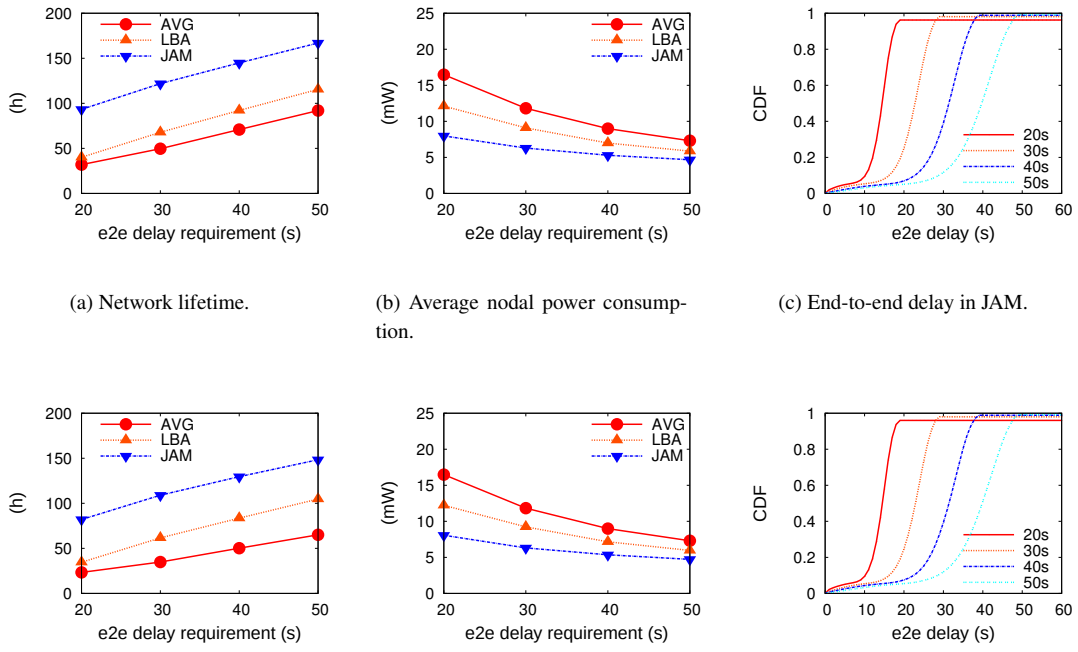
5.5.1 NS-2 Simulations

In the simulation, nodes are randomly deployed in a 500 m $\times$ 500 m area and the sink is located at the center of the area. The evaluation results are averaged over 30 different random topologies. We

vary the end-to-end delay requirement, initial nodal energy distribution, and the network density in the simulation. The default initial nodal energy is 4500 Joules. The maximal communication range is 70 meters and the power consumption is 69 mW when radio is on. All nodes are sources and the data generation rate is a random value between 0.1 and 1 packet per second. The packet size is 128 bytes. In both simulations and testbed experiments, $\delta = 0.5$ s, $\varepsilon = 0.1$ s, $W_{\min} = 1$ minute, and $W_{\max} = 16$ minutes.

5.5.1.1 Performance under Different End-to-End Delay Requirements

Figure 5.8 compares the performances of all the evaluated schemes with the end-to-end delay requirement varying between 20 s and 50 s. Number of nodes in the network is 60, and initial Tr is 1.5 s. First row of Figure 5.8 shows evaluation results when all nodes start with 4500 J energy while second row shows results when the initial nodal energy is a random value between $4500 * (1 \pm 40\%)$ J.



(a) Network lifetime. (b) Average nodal power consumption. (c) End-to-end delay in JAM.

(d) Network lifetime. (e) Average nodal power consumption. (f) End-to-end delay in JAM.

Figure 5.8 Simulation results under different end-to-end delay requirements.

Different from LBA which does not improve the network lifetime much when the initial nodal energy distribution is homogeneous as shown in Figure 5.8(a) (similar finding has also been observed in [71]), JAM consistently improves the network lifetime in both homogeneous and heterogeneous initial energy settings under all end-to-end delay requirements. Specifically, when all nodes start with the same amount of energy, compared with LBA, JAM improves the network lifetime by 158% and 59% when the end-to-end delay requirement is 20 s and 50 s, respectively. When nodes start with different energy, the improvement ratio is 165% and 50% when the requirement is 20 s and 50 s, respectively.

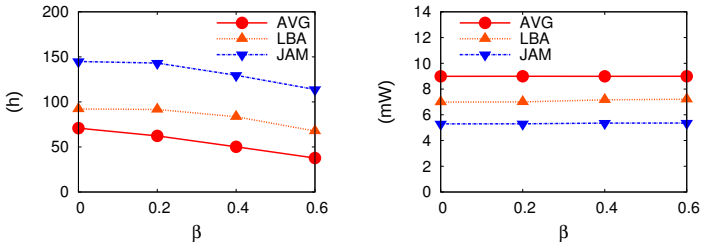
JAM yields more significant network lifetime improvement when the end-to-end delay requirement is more stringent because smaller delay bound means more limited FAD values for each node along source-to-sink paths, which may result in insufficient data suppression and hence heavier network traffic. As MAC behaviors in AVG and LBA are not jointly adjusted with aggregation behaviors, the communication overhead could be expensive. Hence, the heavy traffic could soon deplete the nodal energy and constrain the network lifetime. In comparison, JAM yields a much lower average nodal energy consumption due to its joint MAC and aggregation design, as shown in Figures 5.8(b) and 5.8(e).

Figures 5.8(c) and 5.8(f) plot the CDF (Cumulative Distribution Function) of the end-to-end delay for the JAM scheme. Results show that all the end-to-end delay requirements are well-satisfied. Similar to RI-MAC, JAM drops a data packet after a certain number (4 in our implementation) of failed transmission attempts. Therefore, when the end-to-end delay requirement is relatively small, i.e., 20 s, the heavy traffic deteriorates channel contentions and about 4% of data packets are dropped, while the data delivery ratio approaches 100% when the end-to-end delay requirement increases to 50 s. Due to space limitation, we omit the CDF results of the end-to-end delay for other evaluation scenarios, where JAM exhibits a similar performance as the ones shown in Figures 5.8(c) and 5.8(f).

5.5.1.2 Performance under Different Initial Energy

We now evaluate the impact of the initial nodal energy heterogeneity level (denoted as β). It is defined as follows. With a heterogeneity level of β, the initial nodal energy is a random value between $4500 * (1 \pm \beta)$ Joules. Results are shown in Figure 5.9. Number of nodes in the network is 60,

end-to-end delay requirement is 40 s, and initial Tr is 1.5 s.



(a) Network lifetime.

(b) Average nodal power consumption.

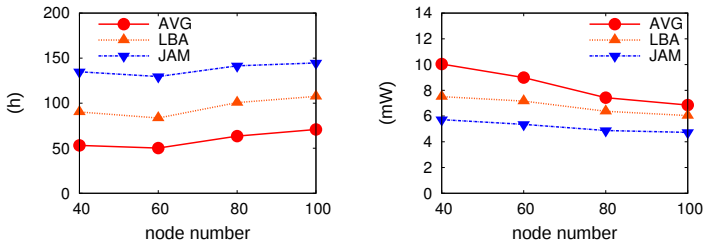
Figure 5.9 Simulation results under different initial energy heterogeneity levels.

With AVG, as Tr and FAD are not adjusted at runtime, network lifetime is bounded by the initial minimal-lifetime node. As the initial energy distribution becomes more and more heterogeneous (i.e., β increases), network lifetime achieved by AVG drops quickly. On the other hand, both LBA and JAM re-distribute the end-to-end delay so that energy bottleneck nodes could be saved by other high-energy nodes along the same route and hence yield a significantly longer network lifetime. As a larger β demands more re-distribution efforts, the network lifetime drops as well with LBA and JAM. However, as all the nodes along the route absorb this effect collaboratively, the network lifetime drops less quickly. For example, network lifetime drops 21% with JAM when β increases from 0% to 60%, in comparison to 46% with AVG. Finally, thanks to the joint MAC and aggregation design, JAM yields a consistently 70% longer network lifetime than LBA under different β values.

5.5.1.3 Performance under Different Densities

As shown in Figure 5.10, JAM always yields a significantly longer network lifetime than other schemes, regardless of the network density. The end-to-end delay requirement is 40 s, initial Tr is 1.5 s, and initial nodal energy level is a random value between $4500 * (1 \pm 40\%)$ J. It is interesting to see (in Figure 5.10(b)) that the average nodal power consumption decreases as the network density increases. This is because, with a higher network density and consequently a higher node degree on average, a source node may reach the sink via a shorter path. This means less nodes are involved in the path and

the overall energy consumption is thus reduced.



(a) Network lifetime.

(b) Average nodal power consumption.

Figure 5.10 Simulation results under different network densities.

5.5.2 Testbed Experiments

We set up a testbed network of 32 TelosB motes, forming a fixed tree topology shown in Figure 5.11. All nodes are sources, and the data generation interval is uniformly distributed between 0.8 and 1.2 s. The default Tr value is 1.5 s, and the end-to-end delay requirement varies between 20 s and 40 s.

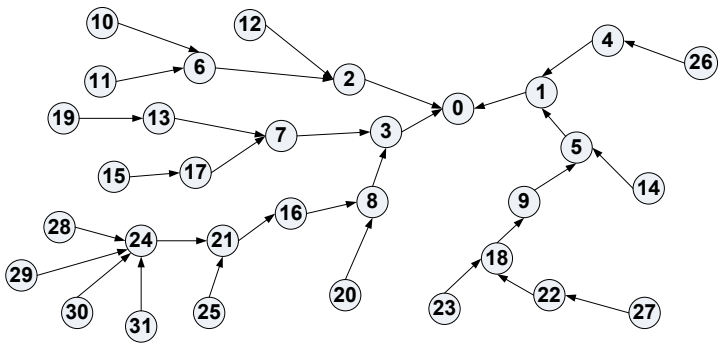
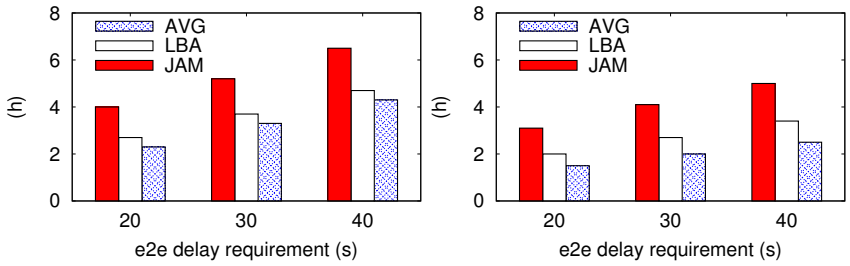


Figure 5.11 Network topology in testbed experiments.

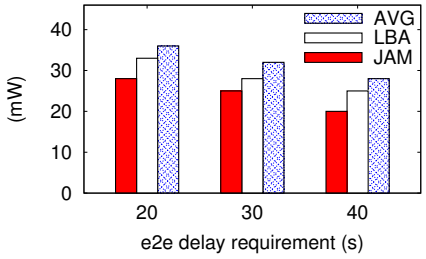
In order to complete the experiments within a reasonable amount of time, *we study how fast a node consumes a small designated amount of energy, and evaluate its nodal lifetime as the time period during which the designated amount of energy is consumed.* We run two sets of experiments. In the first set, the initial available energy distribution is uniform and all nodes have 450 Joules; results are shown in the left column of Figure 5.12. In the second set, the initial available energy at an individual node

is a random value between 250 and 450 Joules; results are shown in the right column of Figure 5.12. Overall, experiment results confirm our observations in the simulation study.

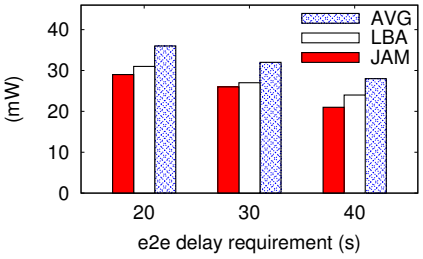


(a) Network lifetime.

(b) Network lifetime.



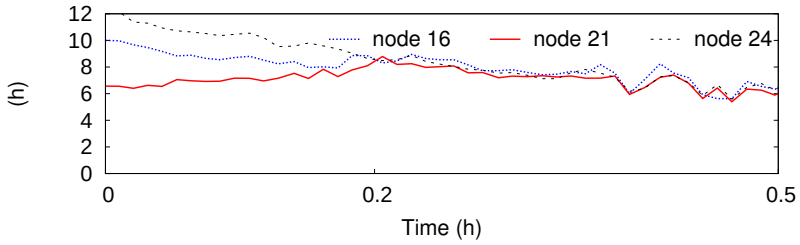
(c) Average nodal power consumption.



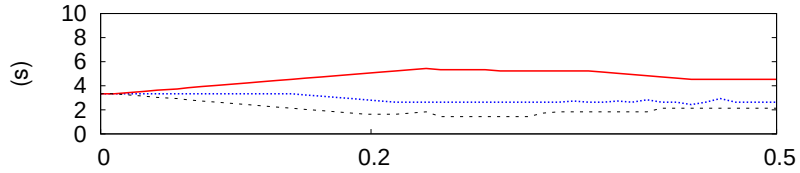
(d) Average nodal power consumption.

Figure 5.12 Experiment results under different end-to-end delay requirements.

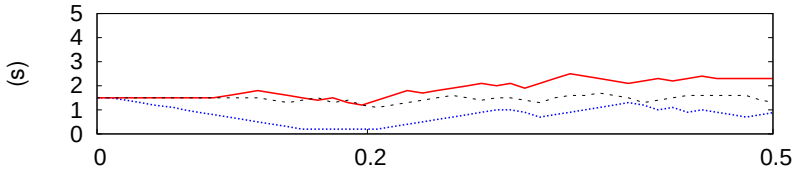
To illustrate how JAM adaptively tunes MAC and aggregation operational parameters to balance nodal lifetime within the neighborhood, we plot in Figure 5.13 the changing traces of the operational parameters of the nodes along the path $24 \rightarrow 21 \rightarrow 16$. We observe that during time period [0, 0.2] h, as shown in Figure 5.13(a), node 21 has a lower nodal lifetime than nodes 16 and 24. To improve node 21's lifetime, (i) node 16, as node 21's parent, decreases its Tr to save node 21's energy cost on idly waiting during transmissions; (ii) node 21 increases its FAD to reduce the amount of outgoing traffic; and (iii) node 24 decreases its Tr to ensure the delay requirement is satisfied. As a result, their nodal lifetimes are balanced gradually. During time period [0.2, 0.5] h, as all three nodes have reached a similar level of nodal lifetime, their operational parameters are relatively stabilized.



(a) Nodal lifetime.



(b) Nodal FAD values.



(c) Nodal Tr values.

Figure 5.13 Changing traces of nodal lifetime, FAD, and Tr values.

5.6 Conclusions

In this study, we present a new holistic approach called JAM to prolong the sensor network lifetime. Different from the existing works which adapt either MAC or aggregation behavior alone, JAM integrates the advantages of both approaches and therefore can extend the network lifetime more effectively. In addition, JAM can also meet the end-to-end delay requirement specified by the applications.

JAM was designed to work with static data collection trees. In the future, we will improve the JAM design by taking into account the routing strategy. As routing behavior also affects the network traffic distribution, a joint MAC, routing, and aggregation design may prolong the network lifetime even further.

CHAPTER 6. CONCLUSIONS AND POSSIBLE FUTURE TOPICS

In this chapter, we summarize the main contributions in this dissertation. Moreover, we discuss several potential research topics related to wireless chargeable sensor networks (WCSNs) based on our past research efforts.

6.1 Research Contributions

In this dissertation, we proposed several practical solutions for WCSN lifetime elongation and sensed data utility maximization. We have demonstrated their effectiveness via extensive experimental and simulation studies. The main contributions of our work are:

- *Efficient Joint Routing and Charging Scheduling to Improve Sensor Network Lifetime*

  In Chapter 2, we study the problem of prolonging network lifetime with a single MC in WCSNs. We propose a practical and efficient scheme, name proposed J-RoC, to improve the network lifetime. Through proactively guiding the routing activities in the network and delivering energy to where it is needed, J-RoC not only replenishes energy into the network but also improves the network energy utilization efficiency. Extensive experimental and simulation studies demonstrate that J-RoC significantly elongates the network lifetime compared with existing wireless charging based schemes.

- *Efficient Joint Charging and Rate Allocation to Maximize Sustainable Sensor Network Utility*

  In Chapter 3, we propose a practical and efficient joint charging and rate allocation scheme, called JCRA, to maximize the network utility while ensuring eternal network lifetime. We present the design and implementation of the JCRA scheme and show its effectiveness in improving the network utility compared with the centralized, off-line solution via ns-2 simulations under

various network configurations.

- *Lifetime Balanced Data Aggregation in Delay-bounded and Energy-Heterogeneous Sensor Networks*

  In Chapter 4, we investigate how to extend energy-heterogeneous sensor network lifetime with a given application-specific end-to-end data delivery delay bound requirement. We propose a data aggregation scheme named LBA. In contrast to existing aggregation schemes that focus on reducing the energy consumption and extending the lifetime of each individual node, LBA has a unique design goal to balance the nodal lifetime and thus prolong the network lifetime more effectively. To achieve this goal in a distributed manner, LBA adaptively adjusts the aggregation holding time between neighboring nodes to balance their nodal lifetime; as such balancing takes place in all neighborhoods, nodes in the entire network can gradually adjust their nodal lifetime towards the globally balanced status. Experimental studies on a sensor network testbed show that LBA can achieve the design goal and approach the theoretical performance upperbound.

- *Efficient Joint Aggregation and MAC Design to Prolong Sensor Network Lifetime*

  In Chapter 5, we present a new holistic approach called JAM to prolong the sensor network lifetime. Different from the existing works which adapt either MAC or aggregation behavior alone, JAM integrates the advantages of both approaches and therefore can extend the network lifetime more effectively. In addition, JAM can also meet the end-to-end delay requirement specified by the applications. Extensive ns-2 simulation and TinyOS experiment results demonstrate the effectiveness of JAM in prolonging the network lifetime compared with the state-of-the-art schemes.

6.2 Possible Future Research Topics

The past research experiences greatly help us understand various problems in WCSNs. In this section, we share some of our opinions on these problems and discuss several potential research topics that are essential for future WCSN research.

- First of all, multiple MCs may be deployed in large scale WCSNs due to the limited charging

capabilities of a single MC. How to efficiently coordinate the charging behaviors of multiple MCs and jointly optimize the charging activities and networking protocol design, pose as practically important research problems.

- Secondly, the broadcast characteristic of wireless charging shall be further investigated. It is known that the efficiency of wireless charging is approximately linearly related to the number of nodes being charged simultaneously [43]. If we intentionally deploy multiple sensor nodes around lifetime bottleneck nodes, e.g., nodes close to the base station, the network lifetime may be improved dramatically. However, as the networks conditions are highly dynamic, it may be hard to predict where the bottleneck nodes will appear. In this case, efficient node redeployment strategies shall be considered.

- Finally, besides data aggregation and MAC protocols, routing behavior also affects the network traffic distribution. Therefore, a joint routing, aggregation and MAC design may better handle the energy heterogeneity problem in WCSNs and prolong the network lifetime even further.

Bibliography

[1] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk, and J. Anderson, "Wireless Sensor Networks for Habitat Monitoring," in *Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications*, ser. WSNA '02, 2002, pp. 88–97.

[2] N. Xu, S. Rangwala, K. K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, and D. Estrin, "A Wireless Sensor Network For Structural Monitoring," in *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*, ser. SenSys '04, 2004, pp. 13–24.

[3] S. H. Lee, S. Lee, H. Song, and H. S. Lee, "Wireless sensor network design for tactical military applications: remote large-scale environments," in *Proceedings of the 28th IEEE Conference on Military Communications*, ser. MILCOM'09, 2009, pp. 911–917.

[4] O. Chipara, C. Lu, T. C. Bailey, and G.-C. Roman, "Reliable clinical monitoring using wireless sensor networks: experiences in a step-down hospital unit," in *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, ser. SenSys '10, 2010, pp. 155–168.

[5] Y. Ma, M. Richards, M. Ghanem, Y. Guo, and J. Hassard, "Air pollution monitoring and mining based on sensor grid in london," *Sensors*, vol. 8, no. 6, pp. 3601–3623, 2008.

[6] J. Chang and L. Tassiulas, "Energy Conserving Routing in Wireless Ad-hoc Networks," in *Proceedings of the 19th Annual Joint Conference of the IEEE Computer and Communications Societies*, ser. INFOCOM '00, 2000, pp. 22–31.

[7] Q. Li, J. Aslam, and D. Rus, "Online Power-Aware Routing in Wireless Ad-Hoc Networks," in *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '01, 2001, pp. 97–107.

[8] A. Kansal, D. Potter, and M. B. Srivastava, "Performance Aware Tasking for Environmentally Powered Sensor Networks," *SIGMETRICS Perform. Eval. Rev.*, vol. 32, no. 1, pp. 223–234, 2004.

[9] K.-W. Fan, Z. Zheng, and P. Sinha, "Steady and Fair Rate Allocation for Rechargeable Sensors in Perpetual Sensor Networks," in *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems*, ser. SenSys '08, 2008, pp. 239–252.

[10] C. Park and P. Chou, "AmbiMax: Autonomous Energy Harvesting Platform for Multi-Supply Wireless Sensor Nodes," in *Proceedings of the 3rd Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, ser. SECON '06, 2006, pp. 168–177.

[11] S. M. Jose, J. O. Mur-mir, R. Amirtharajah, A. P. Ch, and J. H. Lang, "Vibration-to-electric energy conversion," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 9, no. 1, pp. 64–76, 2001.

[12] B. Tong, Z. Li, G. Wang, and W. Zhang, "On-Demand Node Reclamation and Replacement for Guaranteed Area Coverage in Long-lived Sensor Networks," in *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, ser. QShine '09, 2009, pp. 148–166.

[13] B. Tong, G. Wang, W. Zhang, and C. Wang, "Node Reclamation and Replacement for Long-lived Sensor Networks," in *Proceedings of the 6th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, ser. SECON'09, 2009, pp. 592–600.

[14] Powercast, "Online link," http://www.powercastco.com.

[15] A. Kurs, A. Karalis, M. Robert, J. D. Joannopoulos, P. Fisher, and M. Soljacic, "Wireless Power Transfer via Strongly Coupled Magnetic Resonances," *Science*, vol. 317, pp. 83–86, 2007.

[16] Garcia, "Online link," http://www.acroname.com.

[17] L. Su, Y. Gao, Y. Yang, and G. Cao, "Towards optimal rate allocation for data aggregation in wireless sensor networks," in *Proceedings of the 12th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, ser. MobiHoc '11, 2011, pp. 19:1–19:11.

[18] X. Wang and K. Kar, "Cross-layer rate control for end-to-end proportional fairness in wireless networks with random access," in *Proceedings of the 6th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, ser. MobiHoc '05, 2005, pp. 157–168.

[19] X. Jiang, J. Polastre, and D. Culler, "Perpetual Environmentally Powered Sensor Networks," in *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks*, ser. IPSN '05, 2005.

[20] P. Dutta, J. Hui, J. Jeong, S. Kim, C. Sharp, J. Taneja, G. Tolle, K. Whitehouse, and D. Culler, "Trio: Enabling Sustainable and Scalable Outdoor Wireless Sensor Network Deployments," in *Proceedings of the 5th International Symposium on Information Processing in Sensor Networks*, ser. IPSN '06, 2006, pp. 407–415.

[21] F. Zhang, X. Liu, S. Hackworth, R. Sclabassi, and M. Sun, "In Vitro and In Vivo Studies on Wireless Powering of Medical Sensors and Implantable Devices," in *Life Science Systems and Applications Workshop, IEEE/NIH*, ser. LiSSA '09, 2009, pp. 84–87.

[22] Y. Peng, Z. Li, W. Zhang, and D. Qiao, "Prolonging Sensor Network Lifetime Through Wireless Charging," in *Proceedings of the 31st IEEE Real-Time Systems Symposium*, ser. RTSS '10, 2010, pp. 129–139.

[23] S. He, J. Chen, J. Fachang, D. K. Y. Yau, G. Xing, and Y. Sun, "Energy Provisioning in Wireless Rechargeable Sensor Networks," in *Proceedings of the 30th Annual Joint Conference of the IEEE Computer and Communications Societies*, ser. INFOCOM '11, 2011, pp. 2006–2014.

[24] WirelessPowerConsortium, http://www.wirelesspowerconsortium.com/.

[25] Y. Shi, L. Xie, Y. T. Hou, and H. D. Sherali, "On Renewable Sensor Networks with Wireless Energy Transfer," in *Proceedings of the 30th Annual Joint Conference of the IEEE Computer and Communications Societies*, ser. INFOCOM '11, 2011, pp. 1350–1358.

[26] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis, "Collection Tree Protocol," in *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, ser. SenSys '09, 2009, pp. 1–14.

[27] K. Kar, M. Kodialam, T. V. Lakshman, and L. Tassiulas, "Routing for Network Capacity Maximization in Energy-constrained Ad-hoc Networks," in *Proceedings of the 22th Annual Joint Conference of the IEEE Computer and Communications Societies*, ser. INFOCOM '03, 2003, pp. 673–681.

[28] K. Srinivasan, M. A. Kazandjieva, S. Agarwal, and P. Levis, "The Beta-factor: Measuring Wireless Link Burstiness," in *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems*, ser. SenSys '08, 2008, pp. 29–42.

[29] J. Zhao and R. Govindan, "Understanding Packet Delivery Performance in Dense Wireless Sensor Networks," in *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems*, ser. SenSys '03, 2003, pp. 1–13.

[30] K. Dantu, M. Rahimi, H. Shah, S. Babel, A. Dhariwal, and S. G. S., "Robomote: Enabling Mobility in Sensor Networks," in *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks*, ser. IPSN '05, 2005, pp. 404–409.

[31] JOpt.NET, "Online link," http://www.dna-evolutions.com.

[32] J. K. Lenstra and A. H. G. R. Kan, "Complexity of Vehicle Routing and Scheduling Problems," *Networks*, vol. 11, no. 4, pp. 221–227, 1981.

[33] M. Buettner, G. V. Yee, E. Anderson, and R. Han, "X-MAC: a short preamble MAC protocol for duty-cycled wireless sensor networks," in *Proceedings of the 4th International Conference on Embedded Networked Sensor Systems*, ser. SenSys '06, 2006, pp. 307–320.

[34] C. Moser, L. Thiele, D. Brunelli, and L. Benini, "Adaptive Power Management for Environmentally Powered Systems," *Computers, IEEE Transactions on*, vol. 59, no. 4, pp. 478–491, 2010.

[35] R.-S. Liu, P. Sinha, and C. Koksal, "Joint Energy Management and Resource Allocation in Rechargeable Sensor Networks," in *Proceedings of the 29th Annual Joint Conference of the IEEE Computer and Communications Societies*, ser. INFOCOM '10, 2010, pp. 1–9.

[36] B. Zhang, R. Simon, and H. Aydin, "Maximum utility rate allocation for energy harvesting wireless sensor networks," in *Proceedings of the 14th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, ser. MSWiM '11, 2011, pp. 7–16.

[37] Z. Li, Y. Peng, W. Zhang, and D. Qiao, "J-RoC: a Joint Routing and Charging Scheme to Prolong Sensor Network Lifetime," in *Proceedings of the 19th IEEE International Conference on Network Protocols*, ser. ICNP '11, 2011, pp. 373–382.

[38] L. Xie, Y. Shi, Y. T. Hou, W. Lou, H. D. Sherali, and S. F. Midkiff, "Bundling Mobile Base Station and Wireless Energy Transfer: Modeling and Optimization," in *Proceedings of the 32th Annual Joint Conference of the IEEE Computer and Communications Societies*, ser. INFOCOM '13, 2013, pp. 1636–1644.

[39] I. Stark, "Invited Talk: Thermal Energy Harvesting with Thermo Life," in *Proceeding of International Workshop on Wearable and Implantable Body Sensor Networks*, ser. BSN '06, 2006, pp. 19–22.

[40] D. K. Noh and K. Kang, in *Proceedings of 18th Internatonal Conference on Computer Communications and Networks*.

[41] D. K. Noh, L. Wang, Y. Yang, H. K. Le, and T. F. Abdelzaher, "Minimum Variance Energy Allocation for a Solar-Powered Sensor System," in *Proceedings of the 5th IEEE International Conference on Distributed Computing in Sensor Systems*, ser. DCOSS '09, 2009, pp. 44–57.

[42] S. Chen, Y. Fang, and Y. Xia, "Lexicographic Maxmin Fairness for Data Collection in Wireless Sensor Networks," *Mobile Computing, IEEE Transactions on*, vol. 6, no. 7, pp. 762–776, 2007.

[43] B. Tong, Z. Li, G. Wang, and W. Zhang, "How Wireless Power Charging Technology Affects Sensor Network Deployment and Routing," in *Proceedings of the IEEE 30th International Conference on Distributed Computing Systems*, ser. ICDCS '10, 2010, pp. 438–447.

[44] L. Xie, Y. Shi, Y. T. Hou, W. Lou, H. D. Sherali, and S. F. Midkiff, "On Renewable Sensor Networks with Wireless Energy Transfer: The Multi-Node Case," in *Proceedings of the 9th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, ser. SECON '12, 2012, pp. 10–18.

[45] L. Fu, P. Cheng, Y. Gu, J. Chen, and T. He, "Minimizing charging delay in wireless rechargeable sensor networks," in *Proceedings of the 32th Annual Joint Conference of the IEEE Computer and Communications Societies*, ser. INFOCOM '13, 2013, pp. 2922–2930.

[46] L. He, Y. Gu, J. Pan, and T. Zhu, "On-Demand Charging in Wireless Sensor Networks: Theories and Applications," in *Proceedings of the 10th IEEE International Conferece on Mobile Ad-hoc and Sensor Systems*, ser. MASS '13, 2013.

[47] A. Kansal, J. Hsu, S. Zahedi, and M. B. Srivastava, "Power management in energy harvesting sensor networks," *ACM Trans. Embed. Comput. Syst.*, vol. 6, no. 4, 2007.

[48] Ipopt, "Online link," https://projects.coin-or.org/Ipopt.

[49] A. Cammarano, C. Petrioli, and D. Spenza, "Pro-Energy: A novel energy prediction model for solar and wind energy-harvesting wireless sensor networks," in *Proceedings of the 9th IEEE International Conferece on Mobile Ad-hoc and Sensor Systems*, ser. MASS '12, 2012.

[50] Y. Sun, O. Gurewitz and D. Johnson, "RI-MAC: a receiver-initiated asynchronous duty cycle MAC protocol for dynamic traffic loads in wireless sensor networks," in *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems*, ser. SenSys '08, 2008, pp. 1–14.

[51] Z. Li, Y. Peng, D. Qiao, and W. Zhang, "Joint Aggregation and MAC Design to Prolong Sensor Network Lifetime," in *Proceedings of the 21th IEEE International Conference on Network Protocols*, ser. ICNP '11, 2013.

[52] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "TinyDB: An Acquisitional Query Processing System for Sensor Networks," *ACM Trans. Database Syst*, vol. 30, no. 1, pp. 122–173, Mar. 2005.

[53] R. Kumar, M. Wolenetz, B. Agarwalla, J. Shin, P. Hutto, A. Paul, and U. Ramachandran, "DFuse: a Framework for Distributed Data Fusion," in *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems*, ser. SenSys '03, 2003, pp. 114–125.

[54] L. Becchetti, A. Marchetti-Spaccamela, A. Vitaletti, P. Korteweg, M. Skutella, and L. Stougie, "Latency-Constrained Aggregation in Sensor Networks," *ACM Trans. Algorithms*, vol. 6, no. 1, pp. 13:1–13:20, 2009.

[55] W. Wang, V. Srinivasan, and K. C. Chua, "Using Mobile Relays to Prolong the Lifetime of Wireless Sensor Networks," in *Proceedings of the 11th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '05, 2005, pp. 270–283.

[56] J. Chang and L. Tassiulas, "Maximum Lifetime Routing in Wireless Sensor Networks," *IEEE/ACM Trans. Netw.*, vol. 12, no. 4, pp. 609–619, 2004.

[57] K. Fan, S. Liu, and P. Sinha, "On the Potential of Structure-Free Data Aggregation in Sensor Networks," in *Proceedings of the 25th Annual Joint Conference of the IEEE Computer and Communications Societies*, ser. INFOCOM '06, 2006, pp. 1–12.

[58] ——, "Scalable Data Aggregation for Dynamic Events in Sensor Networks," in *Proceedings of the 4th International Conference on Embedded Networked Sensor Systems*, ser. SenSys '06, 2006, pp. 181–194.

[59] T. He, B. M. Blum, J. A. Stankovic, and T. Abdelzaher, "AIDA: Adaptive Application Independent Data Aggregation in Wireless Sensor Networks," in *ACM Trans. Embed. Comput. Syst.*, 2004.

[60] Z. Ye, A. Abouzeid, and J. Ai, "Optimal Policies for Distributed Data Aggregation in Wireless Sensor Networks," in *Proceedings of the 26th Annual Joint Conference of the IEEE Computer and Communications Societies*, ser. INFOCOM '07, 2007, pp. 1676–1684.

[61] I. Solis and K. Obraczka, "The Impact of Timing in Data Aggregation for Sensor Networks," in *Proceedings of the 2004 IEEE International Conference on Communications*, ser. ICC '04, vol. 6, 2004, pp. 3640–3645.

[62] Q. Xiang, J. Xu, X. Liu, H. Zhang, and L. Rittle, "When In-Network Processing Meets Time: Complexity and Effects of Joint Optimization in Wireless Sensor Networks," in *Proceedings of the 30st IEEE Real-Time Systems Symposium*, ser. RTSS '09, 2009, pp. 148–157.

[63] S. Hariharan and N. Shroff, "Maximizing Aggregated Revenue in Sensor Networks under Deadline Constraints," in *Proceedings of the 48th IEEE Conference on Decision and Control*, ser. CDC '09, 2009, pp. 4846–4851.

[64] Y. Wu, S. Fahmy, and N. B. Shroff, "On the Construction of a Maximum-Lifetime Data Gathering Tree in Sensor Networks: NP-Completeness and Approximation Algorithm," in *Proceedings of the 27th Annual Joint Conference of the IEEE Computer and Communications Societies*, ser. INFOCOM '08, 2008, pp. 47–57.

[65] C. Hua and T.-S. P. Yum, "Optimal Routing and Data Aggregation for Maximizing Lifetime of Wireless Sensor Networks," *IEEE/ACM Transactions on Networking*, vol. 16, no. 4, pp. 892–903, 2008.

[66] S. F. Madden, M. J. Hellerstein, and W. J. M. Hong, "TAG: A Tiny AGgregation Service for Ad-Hoc Sensor Networks," in *OPERATING SYSTEMS REVIEW*, 2002, VOL 36.

[67] P. Dutta, S. Dawson-Haggerty, Y. Chen, C.-J. M. Liang, and A. Terzis, "Design and Evaluation of a Versatile and Efficient Receiver-Initiated Link Layer for Low-Power Wireless," in *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, ser. SenSys '10, 2010, pp. 1–14.

[68] R. Braynard, A. Silberstein, and C. Ellis, "Extending Network Lifetime Using an Automatically Tuned Energy-Aware MAC Protocol," in *Proceedings of the 3rd European Conference on Wireless Sensor Networks*, ser. EWSN'06, 2006, pp. 244–259.

[69] Z. Li, M. Li, and Y. Liu, "Towards Energy-Fairness in Asynchronous Duty-Cycling Sensor Networks," in *Proceedings of the 31th Annual Joint Conference of the IEEE Computer and Communications Societies*, ser. INFOCOM'12, 2012, pp. 801–809.

[70] A. Meier, M. Woehrle, M. Zimmerling, and L. Thiele, "ZeroCal: Automatic MAC Protocol Calibration," in *Proceedings of the 6th IEEE International Conference on Distributed Computing in Sensor Systems*, ser. DCOSS'10, 2010, pp. 31–44.

[71] Z. Li, Y. Peng, D. Qiao, and W. Zhang, "LBA: Lifetime Balanced Data Aggregation in Low Duty Cycle Sensor Networks," in *Proceedings of the 31th Annual Joint Conference of the IEEE Computer and Communications Societies*, ser. INFOCOM '12, 2012, pp. 1844–1852.

[72] M. Zimmerling, F. Ferrari, L. Mottola, T. Voigt, and L. Thiele, "pTunes: Runtime Parameter Adaptation for Low-Power MAC Protocols," in *Proceedings of the 11th International Conference on Information Processing in Sensor Networks*, ser. IPSN '12, 2012, pp. 173–184.

[73] J. Zhang, X. Jia, and G. Xing, "Real-time Data Aggregation in Contention-based Wireless Sensor Networks," *ACM Trans. Sen. Netw.*, vol. 7, no. 1, pp. 2:1–2:25, 2010.

[74] L. Xiang, J. Luo, and A. V. Vasilakos, "Compressed Data Aggregation for Energy Efficient Wireless Sensor Networks," in *Proceedings of the 8th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, ser. SECON '11, 2011, pp. 46–54.

[75] AllAboutBatteries, "Online link," http://www.allaboutbatteries.com/Energy-tables.html.