

國立臺灣大學電機資訊學院資訊工程學系

博士論文

Department of Computer Science and Information Engineering

College of Electrical Engineering and Computer Science

National Taiwan University

Doctoral Dissertation

資料傳輸排程之能耗與成本最佳化

Energy and Cost Optimization for

Data Communication Scheduling

The seal of National Taiwan University is a circular emblem. It features a central bell (the 'University Bell') flanked by two traditional Chinese lanterns. Above the bell is a stylized tree. The entire emblem is surrounded by a decorative border. The Chinese characters '國立臺灣大學' (National Taiwan University) are inscribed around the perimeter of the seal.

修丕承

Pi-Cheng Hsiu

指導教授：郭大維 博士

Advisor: Tei-Wei Kuo, Ph.D.

中華民國 九十八 年 六 月

June, 2009

Energy and Cost Optimization for Data Communication Scheduling



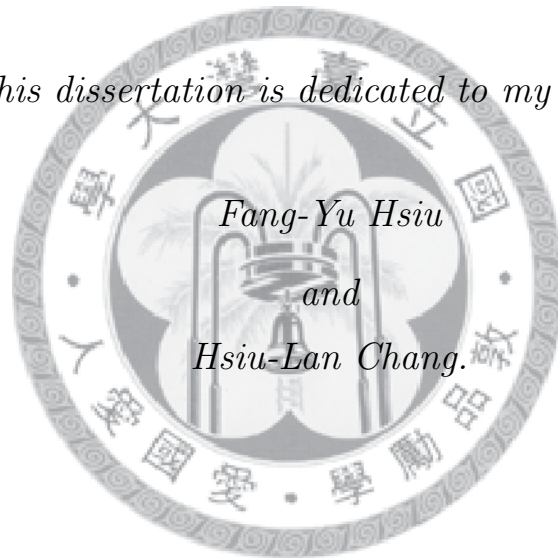
Submitted to
GRADUATE INSTITUTE OF
COMPUTER SCIENCE AND INFORMATION ENGINEERING

In Partial Fulfillment of the Requirements
For the Degree of
DOCTOR OF PHILOSOPHY

At the
NATIONAL TAIWAN UNIVERSITY

June 2009

This dissertation is dedicated to my parents,



*Fang-Yu Hsiu
and
Hsiu-Lan Chang.*

國立臺灣大學博士學位論文
口試委員會審定書

資料傳輸排程之能耗與成本最佳化

Energy and Cost Optimization for Data Communication
Scheduling

本論文係修丕承君（學號D93922014）在國立臺灣大學資訊工程學系完成之博士學位論文，於民國 98 年 4 月 1 日承下列考試委員審查通過及口試及格，特此證明

口試委員：

孫大川

(指導教授)

Lui Sha

龍志昇

逄慶君

洪士濂

楊頌華

張嘉新

陳銘憲

呂育道

系主任

中文摘要

過去十年，高能效與低成本一直是消費性電子產品設計上相當熱門的議題。有鑑於裝置之間以及裝置內部資料傳輸的快速成長，本論文針對可攜式裝置的傳輸元件進行能耗與成本最佳化之研究。

傳輸子系統方面之研究，著重於設計路由協定以達到剩餘電量之最大化。提出多項式時間之最佳演算法對群播路由做最小剩餘電量之最大化。證明最大化叢聚路由之最小剩餘電量為 \mathcal{NP} -hard之問題，且除非 $\mathcal{P} = \mathcal{NP}$ ，否則其對應之最小化問題不存在優於2倍之近似演算法。再針對群播路由，提出分散式演算法及其實踐之路由協定。該協定中，群播路由樹之形成是依照網路中各裝置獨立自主之決定，不需仰賴事先收集整個網路路由資訊。所形成之路由樹被證明不具迴路，且理論上能夠最大化最小剩餘之電量。該協定實作於NS2進行效能評估，結果顯示該協定於各項重要評估指標皆具優越之效能。

傳輸架構方面之研究，在於提出理論之方法以達到匯流排層數之最小化。探討具鏈式優先次序限制之即時工作於多層匯流排系統上最小化傳輸成本之排程問題。首先證明該問題為 \mathcal{NP} -hard，且除非 $\mathcal{P} = \mathcal{NP}$ ，否則不存在優於1.5倍之近似演算法。針對考慮單一多層匯流排以及單位執行與傳輸時間之子問題，提出多項式時間之最佳演算法。再衍生該方法為擬似多項式時間之最佳演算法解決通例問題，以考慮多個多層匯流排、任意執行與傳輸時間、以及不同之時間限制與目標函式。並基於AMBA之系統拓撲，比較最佳演算法與其他啟發式演算法之效能以提供更多有助於系統設計開發之觀點。

關鍵詞：高能效、低成本、路由協定、排程演算法、資料傳輸、網路嵌入式系統

Abstract

Energy- and cost-efficiency designs in consumer electronics have been active research topics in the past decades. This dissertation is highly motivated by the rapid growth of data exchanges for both inter-device and intra-device communication, thus addressing energy conservation and cost reduction by targeting the communication components of portable devices.

The study on communication subsystems aims at the design of a routing protocol for residual-energy maximization. A polynomial-time optimal algorithm is proposed for the multicast case. The aggregate case is proved to be \mathcal{NP} -hard and, unless $\mathcal{P} = \mathcal{NP}$, its minimization version cannot be approximated within a ratio better than 2. A distributed algorithm and its realization, referred to as the Maximum-Residual Multicast Protocol (MRMP), are then developed. In MRMP, a transient multicast tree is derived based on the autonomous decisions of devices in the network, where no global information needs to be collected a priori. The derived tree is proved to be loop-free and theoretically optimal in the maximization of minimum residual energy. The capability of MRMP was evaluated over NS2, for which we have very encouraging results in essential performance metrics adopted for routing protocol evaluation.

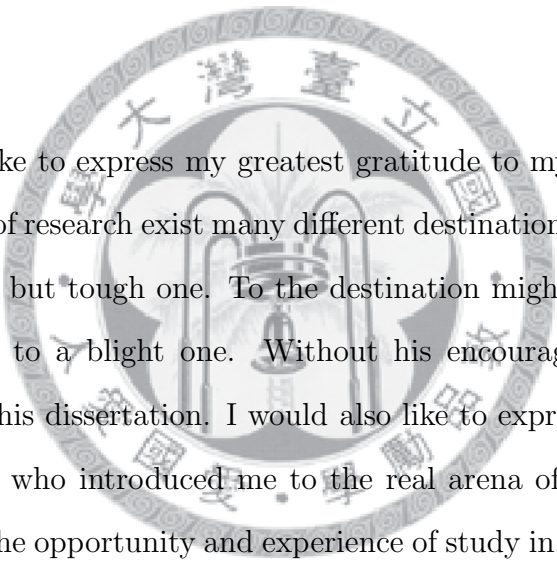
The study on communication architectures focuses on the proposing of a theoretical methodology for bus-layer minimization. Real-time tasks with chain-based precedence constraints are explored on multi-layer bus systems with an objective to minimize the communication cost. The target problem is proved to be \mathcal{NP} -hard and, unless $\mathcal{P} = \mathcal{NP}$, it cannot be approximated within a ratio better than 1.5. A polynomial-time optimal algorithm is first proposed for a restricted case in which one multi-layer bus, and unit execution and communication time are considered. The result is then extended as a pseudo-polynomial-time optimal algorithm in the considerations of multiple multi-layer buses, arbitrary execution and communication time, and different timing constraints and objective functions. The capability of the proposed algorithm was evaluated over an AMBA-like system topology to provide more insights in system designs, compared to some popular heuristics.

Keywords: Energy efficiency, cost efficiency, routing protocols, scheduling algorithms, data communication, networked embedded systems

Acknowledgment

“Research is a repeat search process to find out the unknown.”

-Lui Sha



I would like to express my greatest gratitude to my advisor, Prof. Tei-Wei Kuo. On the way of research exist many different destinations. He always encourages me to pick a vital but tough one. To the destination might lead several roads. He always guides me to a blight one. Without his encouragement and guidance, I would not finish this dissertation. I would also like to express my sincere gratitude to Prof. Lui Sha, who introduced me to the real arena of research field. I would much appreciate the opportunity and experience of study in the University of Illinois at Urbana-Champaign.

I would like to show my appreciation to Prof. Jane W.S. Liu, Prof. Ai-Chun Pang, Prof. Chi-Sheng Shih, and Prof. Jian-Jang Huang for all the skills that I learnt from them during the cooperation in the industrial and research projects. I would also appreciate the fruitful achievements produced from the projects.

I must offer the best thanks to my dissertation committee, Prof. Lui Sha, Prof. Ming-Syan Chen, Prof. Stephen J.H. Yang, Prof. Ai-Chun Pang, Prof. Chi-Sheng Shih, Prof. Shih-Hao Hung, and Dr. Ted Chang. Their valuable comments and suggestions make this dissertation more solid and complete.

I want to thank all the members in the Embedded Systems and Wireless Networking (NEWS) Laboratory. Special thanks go to Yuan-Hao Chang and Yung-Feng Lu for cheering one another on in times of need in the Ph.D program. Special thanks also go to Chin-Hsien Wu and Der-Nien Lee for their discussion and implementation on my research work. Many enjoyments were brought in cooperation with them. My exclusive blessings are given to Chuan-Yue Yang, Pei-Lun Suei, Po-Chun Huang, and Che-Wei Chang for their remaining Ph.D. lives. I also want to thank all the friends in the University of Illinois at Urbana-Champaign. Special thanks go to Ming-Young Nam, Mu Sun, and Chin-Fei Cheah for their particular help in both the research and daily life. I am very sorry for not having space to list all your names and deeply appreciative of all your help.

I am grateful to National Science Council, Ministry of Education, and Yen Tjing Ling Industrial Research Institute for their travel grants to participate in the international conferences, and to my nation for the loans to pursue my Ph.D. degree.

Particularly pay my most gratefulness to my parents, Fang-Yu Hsiu and Hsiu-Lan Chang, and my younger sister, Tzy-Yi Hsiu. Without their support and consideration, I might not sustain the pressure, tide over the difficulties, and finish the degree. This dissertation is dedicated to them.

My special thanks to Pei-Shih Liu for her company in the past five years are beyond works.

Pi-Cheng Hsiu@NTU

June, 2009

Contents

Abstract in Chinese	iv
Abstract	v
Acknowledgment	vi
Contents	viii
List of Figures	xii
List of Tables	xiii
1 Introduction	1
1.1 Motivations	2
1.1.1 Energy-Efficient Routing	2
1.1.2 Cost-Efficient Scheduling	3
1.2 Objectives and Contributions	4
1.2.1 Residual-Energy Maximization	4
1.2.2 Bus-Layer Minimization	6
1.3 Organization	7
2 Related Work	9
2.1 Energy-Efficient Routing	10



2.2	Cost-Efficient Scheduling	14
3	Residual-Energy Maximization	17
3.1	Network Model and Problem Formulation	18
3.2	An Optimal Algorithm for Maximum-Residual Multicasting	21
3.3	A $(2 - \epsilon)$ -Inapproximability Result for Maximum-Residual Aggregating	24
3.4	Performance Evaluation	28
3.4.1	Algorithms for Comparison	28
3.4.2	Experimental Setups and Performance Metrics	29
3.4.3	Experimental Results	31
3.5	Summary	35
4	Distributed Residual-Energy Maximization	36
4.1	Network Model and Problem Formulation	37
4.2	A Distributed Optimal Algorithm for Maximum-Residual Multicasting	40
4.2.1	Algorithm Description	40
4.2.2	Properties	44
4.3	A Maximum-Residual Multicast Protocol	50
4.3.1	Routing Tables	51
4.3.2	Route Discovery	53
4.3.3	Route Establishment	55
4.3.4	Data Forwarding	57
4.4	Performance Evaluation	58
4.4.1	Experimental Setups and Performance Metrics	58
4.4.2	Experimental Results	63
4.5	Summary	68
5	Bus-Layer Minimization	69

5.1	System Model and Problem Formulation	70
5.1.1	System Model	70
5.1.2	Problem Definition	72
5.2	A $(1.5 - \epsilon)$ -Inapproximability Result	74
5.3	A Dynamic-Programming Approach	77
5.3.1	A Basic Algorithm	77
5.3.2	Properties	80
5.4	Extension of the Basic Algorithm	82
5.4.1	Arbitrary Execution and Communication Time	82
5.4.2	Multiple Multi-Layer Buses and Non-Preemptive Tasks/Transactions	86
5.4.3	Different Timing Constraints and Objective Functions	87
5.5	Performance Evaluation	89
5.5.1	Experimental Setups and Performance Metrics	89
5.5.2	Experimental Results	91
5.6	Summary	96
6	Concluding Remarks	97
6.1	Conclusion	97
6.2	Future Research Directions	100
6.2.1	Residual-Energy Maximization	100
6.2.2	Bus-Layer Minimization	101
	Bibliography	102
	Curriculum Vitae	113
	Publication List	114

List of Figures

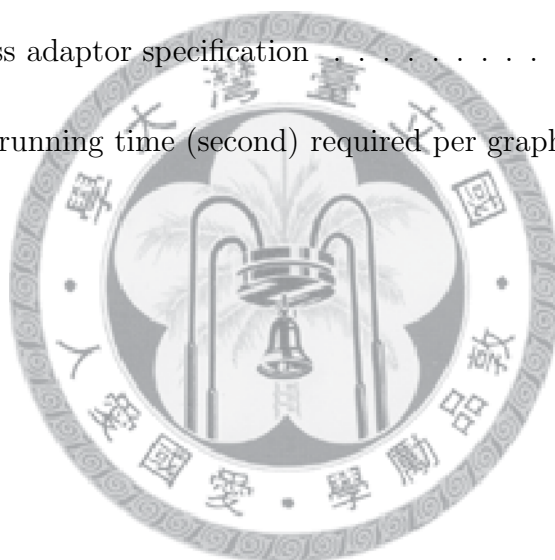
2.1	A simple ad hoc network	12
2.2	The multiple bus and multi-bus architectures	15
3.1	A network topology example	19
3.2	An input instance example	25
3.3	Impacts of the network size ($\gamma = 1\% \times 40^3, \Delta = 10 \times 10^3$)	32
3.4	Impacts of the energy consumption for reception ($ V = 50, \Delta = 10 \times 10^3$)	33
3.5	Impacts of notification thresholds ($ V = 50, \gamma = 1\% \times 40^3$)	34
4.1	The execution of MRMA on an example network	42
4.2	The Maximum-Residual Multicast Protocol (MRMP)	51
4.3	An F-shaped antenna	61
4.4	Network Lifetime	63
4.5	Delivery Ratio	65
4.6	Control Overhead	66
4.7	Propagation Delay	67
5.1	The multi-layer bus architecture	71
5.2	A reduction example	74
5.3	An illustration of terminology	78

5.4	Splitting and grouping of a given precedence graph	83
5.5	Average number of layers needed by each bus	91
5.6	Impacts of the relative weight of buses	93
5.7	Preemptive task/transaction model vs. non-preemptive task/transaction model	94



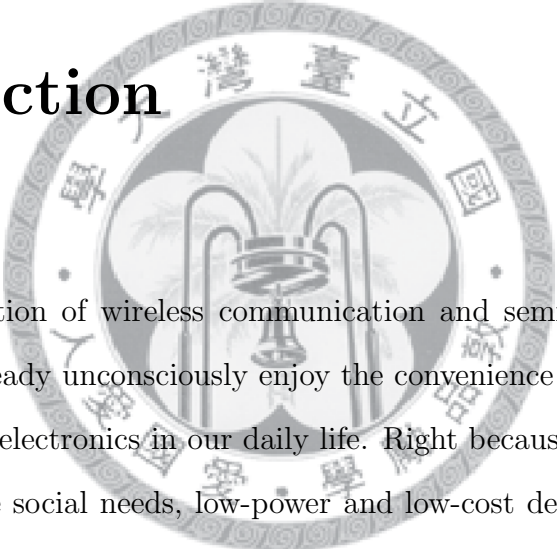
List of Tables

4.1	A wireless adaptor specification	60
5.1	Average running time (second) required per graph	95



Chapter 1

Introduction



With the maturation of wireless communication and semiconductor technologies, human beings already unconsciously enjoy the convenience and interest brought by various consumer electronics in our daily life. Right because of the significant driving force from the social needs, low-power and low-cost design issues in embedded systems have been attractive and active research topics in the decade. Energy conservation and cost reduction are critical since portable devices are always operated with limited battery and usually optimized for specific functionality. Several technologies for energy conservation and cost reduction are being developed by targeting specific components of the portable devices, such as the MPU/DSP and transceiver [41]. The current technologies have gone a long way in this direction and the development appears to continue. Energy-efficient task scheduling and data communication are important topics in this area for limited resource management and arrangement. This dissertation is motivated by the rapid growth of data exchanges for both inter-device and intra-device communication, thus addressing optimization problems of energy consumption and cost from the perspective on data communication.

1.1 Motivations

Routing is the process in which a route from some nodes to others is derived and achieved for data communication in the networking system. Routing protocols designed for autonomous networking systems of portable devices must consider energy consumption of the devices in the network as a primary objective. Power-aware routing metrics serve a heuristic in deriving routes with the best energy efficiency. Identifying an appropriate routing metric, designing a routing algorithm for this metric, and implementing it with a routing protocol are essential problems in data communication. Communication architecture plays another important role in data communication among individual nodes in on-chip systems. Cost has been a major concern in the development of portable devices. Cost-efficient scheduling is essentially a critical problem in minimizing system cost so as to meet performance requirements at the system design stage.

1.1.1 Energy-Efficient Routing

An *ad hoc network* is an autonomous system of wireless devices, known as nodes, connected by wireless links, where packets are transmitted via intermediate nodes, instead of an established infrastructure. The energy consumption required for a transmitter usually increases dramatically with the distance, and that for a receiver is considered as a constant [10, 39]. Because such a node is usually battery-powered, there is a strong demand in power-aware routing. With the popularity of mobile devices, routing becomes increasingly challenging because of the dynamic nature of network topologies and critical energy-efficiency considerations. The problem is further complicated by the existence of a huge population of devices and the needs

in good communication bandwidth utilization, such as the replacement of multiple unicasts with a multicast. Example applications are advertisement in shopping malls [58], tourist information distribution [16], taxi dispatching [34], and cooperative congestion monitoring [59, 78].

1.1.2 Cost-Efficient Scheduling

As the number of data processing engines per system grows significantly in the next few years, how to resolve the communication problem among tasks has become a very challenging and important design issue. The overheads introduced by data exchanging among tasks could easily offset the abundant computing power introduced by many processing engines if related system design issues are not addressed carefully. As powerful system architectures are proposed to resolve the overhead problem, the design issues ironically become even more difficult because the complexity in real-time task scheduling grows significantly. Designs based on experience and/or empirical study could not provide rigid theoretic ground in the minimization of system design cost. Such an observation motivates this work on the proposing of optimal algorithms to minimize the bus cost of real-time embedded systems with multi-layer buses and precedence constraints. For the rest of this dissertation, we shall use terminologies cores, data processing engines, processing elements, and processors interchangeably when there is no ambiguity.

1.2 Objectives and Contributions

This dissertation targets energy-efficiency and low-cost issues in embedded system designs. We are interested in the components for data communication: (1) communication subsystems and (2) communication architectures. Our study on communication subsystems aims at not only the identification of the asymptotical hardness and performance of various power-aware routing metrics, but also the development of a distributed routing protocol adaptable to network topologies and resources that might change over time. For communication architectures, we focus on the proposing of a generic and theoretical methodology for performance/cost exploration on multi-layer bus systems.

1.2.1 Residual-Energy Maximization

The concept of *Maximum-Residual Routing* was first raised by in [72, 73], where the minimum residual energy of nodes is maximized for each multicast. The objective is to prolong the first node failure time when network topologies and data traffic may change frequently in an unpredictable way. In this dissertation, Maximum-Residual Multicasting and Aggregating are explored in heterogeneous wireless ad hoc networks. We propose a *Prim*-like algorithm for Maximum-Residual Multicasting and prove its optimality when up-to-date topology and energy information are available. The proposed algorithm can be applied to many existing routing protocols, especially those based on the link-state approach, *e.g.*, [28, 35]. In protocol designs, control messages for maintaining up-to-date topology and energy information need additional energy consumption. We show by experiments that the proposed algorithm for Maximum-Residual Multicasting intends to find the best route in a

dynamic fashion to prolong network lifetime. The experimental results show that it is excellent in the improvements of network lifetime and load balance, in comparison with other routing metrics, *e.g.*, [50, 72, 77, 85]. On the other hand, we show that Maximum-Residual Aggregating is \mathcal{NP} -hard and approximation algorithms for this problem are unlikely to exist. We then prove that, unless $\mathcal{P} = \mathcal{NP}$, its minimization version cannot be approximated in polynomial time within a ratio of $(2 - \epsilon)$ for any $\epsilon > 0$, where the ratio bound is with respect to the maximum remaining energy before routing.

In protocol designs, the maintenance of global routing information is highly challenging because of the dynamic nature of network topologies and energy resources. The problem is exaggerated when applications with a huge population of mobile devices are under consideration. We first propose a distributed algorithm for *Maximum-Residual Multicast* and prove its optimality without the considerations of node movements and control overheads. When mobility and control message collisions are taken into consideration, it is shown that every derived route remains loop-free and converges toward an optimal solution in the maximization of the minimum residual energy. Based on the proposed algorithm, we then develop a source-initiated on-demand routing protocol, referred to as *Maximum-Residual Multicast Protocol* (MRMP), which is adaptable to network topologies and resources that may change over time. In MRMP, no periodic control message is employed to collect routing information a priori or repair link breakages. Neither group membership nor neighbor relationship is maintained at a node by explicit control messages. When desiring a route, a source invokes a route-discovery procedure over the network, and the individual decisions of intermediate nodes form a loop-free multicast tree naturally. For the performance evaluation, the protocol was implemented over NS2 [26], and simulations were conducted extensively with parameters set based

on a realistic commercial wireless device [2]. We have very encouraging results in essential performance metrics adopted generally for routing protocol evaluation [22].

1.2.2 Bus-Layer Minimization

The *multi-layer bus architecture* provided by ARM [1] further improves the communication concurrency and flexibility [54]. The increasing number of bus layers implies the overheads of the cost per area, power consumption and design complexity. In this dissertation, the problem of scheduling real-time tasks with *chain-type precedence constraints* is explored over multi-layer bus systems with an objective to minimize the bus cost, referred to as the *tardiness-bounded layer minimization problem*. The objective is to minimize the total cost contributed by the needed bus layers without any violation of timing constraints. The contributions of this work start with fundamental but negative results on the \mathcal{NP} -hardness of the target problem and its inapproximability ratio. To be more specific, we show that, unless $\mathcal{P} = \mathcal{NP}$, it is not possible to have any approximation algorithm with an approximation ratio better than 1.5. A polynomial-time optimal algorithm, based on dynamic programming, is then proposed for a restricted case, when tasks only have unit execution times, and any communication delay is of one time unit. The algorithm is later extended as a pseudo-polynomial-time algorithm for general cases when tasks have arbitrary execution times, communication delay can be of any time units, any selected task and/or communication can be preemptive/non-preemptive, and other timing constraints/objective functions are considered. The proposed algorithm was evaluated against a list-scheduling algorithm [75], over an AMBA-based system topology [53] with task generated by TGFF [80], to provide better insights to this work.

1.3 Organization

The rest of this dissertation is organized as follows:

Chapter 2 provides background information on energy-efficiency issues in data communication and reviewed literature related to this dissertation.

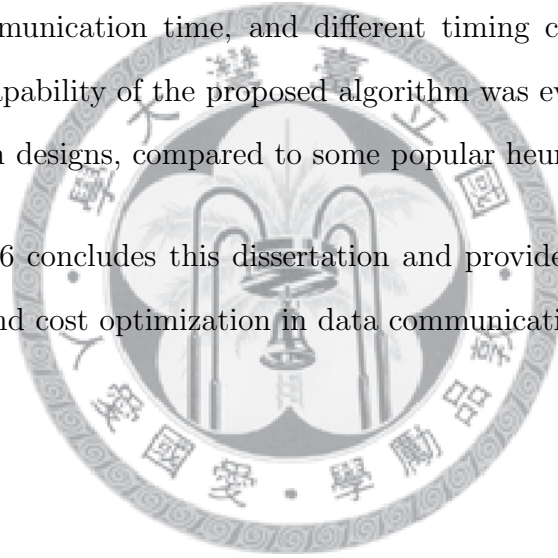
Chapter 3 targets the asymptotical hardness of various power-aware routing metrics and the comparison of their performance. In this chapter, a polynomial-time optimal algorithm is proposed for Maximum-Residual Multicasting. It is then shown that Maximum-Residual Aggregating is \mathcal{NP} -hard and that, unless $\mathcal{P} = \mathcal{NP}$, its minimization version cannot be approximated within a ratio of $(2 - \epsilon)$ for any $\epsilon > 0$. The performance of the routing metric was evaluated by a series of experiments, for which it demonstrated itself being effective and efficient in network lifetime and load balance, in comparison with other routing metrics.

Chapter 4 focus on the development of a routing protocol adaptable to dynamic network topologies and resources. In this chapter, a distributed algorithm and its realization, referred to as MRMP, are proposed for residual-energy maximization. In MRMP, a transient multicast tree is established on demand and derived based on the autonomous decisions of intermediate nodes. The derived tree is proved to be loop-free and theoretically optimal in the maximization of minimum residual energy. The performance of the MRMP was evaluated over NS2 with a series of simulations, for which we have very encouraging results in essential performance metrics adopted generally for routing protocol evaluation.

Chapter 5 aims at the proposing of a theoretical methodology for tackling communication cost optimization for tasks with performance requirements and

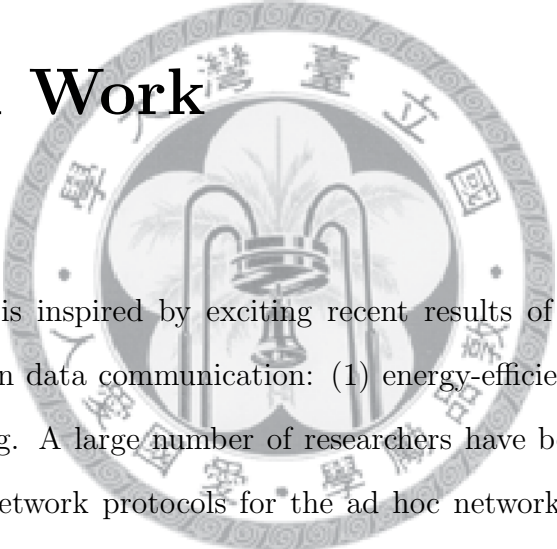
precedence constraints. The tardiness-bounded layer minimization problem is explored in embedded systems with multi-layer buses. In this chapter, we show the \mathcal{NP} -hardness of the problem and the best possible approximation ratio of approximation algorithms. A polynomial-time optimal algorithm is first proposed for a restricted case in which one multi-layer bus, and unit execution and communication time are considered. The result is then extended as a pseudo-polynomial-time optimal algorithm in the considerations of multiple multi-layer buses, arbitrary execution and communication time, and different timing constraints and objective functions. The capability of the proposed algorithm was evaluated to provide more insights in system designs, compared to some popular heuristics.

Chapter 6 concludes this dissertation and provides further research directions in energy and cost optimization in data communication.



Chapter 2

Related Work

The logo of National Taiwan University is a circular emblem. It features a central design with a stylized plant or flower motif. The text "國立台灣大學" (National Taiwan University) is written in Chinese characters around the perimeter of the circle. There are also smaller characters and symbols within the emblem, including a central figure that resembles a traditional lamp or a similar object.

This dissertation is inspired by exciting recent results of two research topics for energy-efficiency in data communication: (1) energy-efficient routing and (2) cost-efficient scheduling. A large number of researchers have been dedicated to power-aware design of network protocols for the ad hoc networking environment. They share a common objective to reduce energy consumption for communication subsystems and, consequently, prolong network lifetime. The main focus of the first part in this chapter is on the survey of various power-aware routing metrics and protocols for wireless ad hoc networks with special attention to Maximum-Residual Routing. The second part reviews literature related to the problems and technologies on cost-efficient scheduling. With the appearance of emerging communication architectures, this research topic has been attractive and active in the decades and many research results over various communication architectures have been proposed in the literature.

2.1 Energy-Efficient Routing

The study on energy-efficient routing can be classified into static or dynamic routing. In static routing, a static network topology is considered, and the data traffic is assumed to be known a priori, *e.g.*, [13, 14, 39, 62, 69]. A well-known example problem is *Maximum-Lifetime Routing*, where packets are routed according to a pre-determined routing plan until the energy of some node drains away. In dynamic routing, both network topologies and the data traffic may change dynamically in an unpredictable way, and nodes may play the role as a source in some dynamic way. In such a routing problem, we have no knowledge on future arrivals. An optimal route is determined on demand based on the network status (*e.g.*, network topology and battery information) at the time being, when a source has packets to route.

In the past decade, various dynamic power-aware routing metrics have been proposed for the prolongation of network lifetime, and a class of fundamental optimization problems were defined, *e.g.*, [50, 72, 73, 77, 85]. Among those routing metrics, *Minimum-Energy Routing* is proposed to minimize the total energy consumption in packet routing. Minimum-Energy Routing can be explored in terms of unicasting or multicasting. Minimum-Energy Unicasting is polynomial-time solvable [73]. However, the minimization of the total energy consumption may result in the rapid energy exhaustion of some specific nodes. In order to avoid such a problem, researchers started proposing algorithms to explore Minimum-Energy Unicasting on a sub-network that excludes nodes with remaining energy lower than a designated threshold (*e.g.*, *Conditional Max-Min battery Capacity Routing* [77]). Another example approach is *Max-Min $\mathcal{Z}P_{min}$ Routing* [50], which derives a routing path by avoiding to route packets via low-energy nodes, where the total energy consumption is at most \mathcal{Z} times that of Minimum-Energy Unicasting.

Multicasting and *aggregating* are techniques suggested to solve the implosion and overlap problems in packet routing. They attempt to minimize the number and the size of duplicate packets and thus to reduce energy consumption. Multicasting transmits packets from a source to multiple destinations by using a group address for the destinations. Aggregating fuses packets coming from multiple destinations enroute to a source. Minimum-Energy Multicasting and Minimum-Energy Aggregating proved to be \mathcal{NP} -hard [20, 42]. Some theoretical analysis was presented in [7, 12, 17, 20, 42, 84], and heuristic algorithms were proposed in different variations. Example results include the minimization of the maximum energy consumption of nodes for routing packets [72], the minimization of the maximum energy consumption of a path for forwarding packets to/from one of the destinations [85], and the maximization of remaining energy *before* packets are routed [50, 77]. They will be introduced in more detail and were adopted for performance comparison in Section 3.4.

In order to prolong network lifetime, the data packets should be routed so that the energy consumption is balanced among the nodes in proportion to their remaining energy, instead of routing to minimize the total energy consumption [15]. Therefore, keeping the minimum remaining energy of nodes as high as possible appears to be a more applicable routing metric. In this dissertation, we explore the maximization of the minimum remaining energy of all nodes *after* packets are routed. The closest related work is that on the maximization of the minimum remaining energy (or a function relative to remaining energy) *before* packets are routed [50, 77]. We use Figure 2.1 to illustrate the difference between the maximization of the minimum remaining energy of nodes *before* and *after* routing. The gray areas in the figure represent the antenna patterns of four heterogeneous nodes with their communication ranges shown. Let the budget functions $\beta()$ and $\hat{\beta}()$ respectively

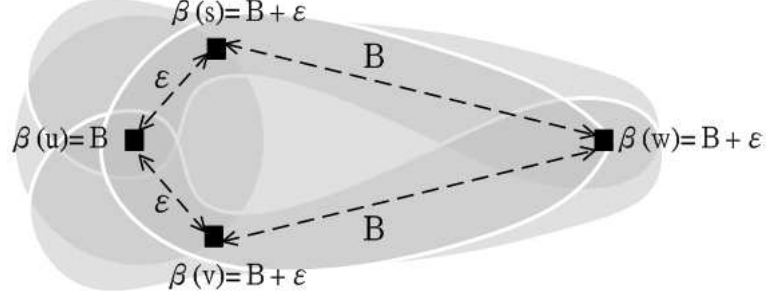


Figure 2.1: A simple ad hoc network

represent the remaining energy of each node *before* and *after* routing. Assume that the weight on each edge represents the energy consumed for the corresponding transmission and that the energy consumption for each reception is ε , where $\varepsilon \ll \mathcal{B}$. Consider the two routing paths $P_1 = s \rightsquigarrow w \rightsquigarrow v$ and $P_2 = s \rightsquigarrow u \rightsquigarrow v$ for node s to send a packet to v . The minimum remaining energy of nodes before routing on P_1 and P_2 is $\beta(s) = \beta(w) = \beta(v) = \mathcal{B} + \varepsilon$ and $\beta(u) = \mathcal{B}$ respectively. When the maximization of the remaining energy *before* routing is considered, P_1 is chosen for routing. As a result, $\hat{\beta}(w) = 0$. However, if the maximization of the remaining energy *after* routing is considered, then P_2 should be picked because the minimum remaining energy (among all nodes) *after* routing is $\hat{\beta}(w) = 0$ and $\hat{\beta}(u) = \mathcal{B} - 2\varepsilon$ via P_1 and P_2 , respectively. In this example, the number of packets from s to v (under the metric on the maximization of the remaining energy *after* routing) is $\frac{\mathcal{B}}{2\varepsilon}$ times that when the maximization of the remaining energy *before* routing is considered.

The maximization problem of the minimum remaining energy among all nodes *after* routing is referred to as *Maximum-Residual Routing*. The idea of Maximum-Residual Routing was first proposed in [73]; however, designing a routing algorithm for this metric and implementing it in a routing protocol were not addressed in the paper. This routing metric is intuitively believed to be effective for prolonging network lifetime [63, 72, 73, 91], but has not been widely studied yet, in

contrast with Minimum-Energy Routing and Maximum-Lifetime Routing. In [88], a *Dijkstra*-like heuristic algorithm for Maximum-Residual Unicasting was proposed to demonstrate its effectiveness in prolonging network lifetime. In [56], a *Dijkstra*-like algorithm for Maximum-Residual Broadcasting was proposed and, in particular, its optimality was also proved when energy consumption for receivers of nodes is the same, *i.e.*, homogeneous wireless ad hoc networks. However, an ad hoc network is usually composed of various mobile devices and not necessarily all nodes desire to be destinations of a session. In addition, they are essentially algorithms relying on the knowledge of the entire topology and the remaining energy information of all nodes, which is highly challenging in routing protocol design.

In the past decades, many excellent routing protocols have been proposed for mobile ad hoc networks, *e.g.*, [5, 23, 48]. Each of them tried to optimize some routing and performance metrics for different application scenarios. Popular metrics include the propagation delay and the delivery ratio, where many studies target applications similar to multiplayer online gaming and teleconferencing [23, 83]. In order to save the network bandwidth, multicast protocols were also widely explored, *e.g.*, [18, 27, 30, 36, 37, 47, 66, 86, 87]. Among those excellent solutions, MAODV [66], ODMRP [47], and DDM [37] are examples of the best ones and were submitted to the IETF MANET Working Group as candidates for standardization. MAODV discovers tree-based routes on demand using a broadcast route-discovery mechanism. MAODV is sensitive to node mobility because it actively tracks and reacts to changes in routes so as to repair link breakages [66]. ODMRP is a mesh-based protocol that provides alternative paths to adapt to topology changes. Control messages are flooded periodically to refresh group membership and update routes, and redundant routes are exploited for data delivery, which makes ODRMP scale not well with network sizes [47]. In DDM, each source is responsible for the maintenance of each

multicast group. The list of destinations is placed in packet headers for self-routing over an underlying unicast protocol. DDM is meant for small multicast groups operating in dynamic networks of any size [37].

Routing over mobile ad hoc networks is complicated by the considerations of energy efficiency [15, 50, 72, 73, 77], while shortest paths are not favored in routing. Many approaches were presented in the literature. However, most of the existing results rely on the knowledge of certain global information, such as the remaining energy of all nodes and/or the minimum transmission power between every pair of nodes. The maintenance problem of similar global information is highly challenging in protocol designs because of the difficulty and cost in the maintenance of up-to-date information. As a result, various assumptions, such as static network topologies and/or fixed traffic patterns, are made to reduce the problem complexity in power-aware routing.

2.2 Cost-Efficient Scheduling

In the last decades, researchers have been exploring architecture designs and task scheduling methodologies for multi-core systems [19, 93]. One major and classical model is the *fully connected architecture*, where the communication between every two processors goes through a dedicated bus, and bus contention is ignored [25, 81, 82, 89]. Due to the challenges on the scalability and cost, the *multiple bus architecture* was proposed, where a set of processors is connected by a collection of buses [60], as shown in Figure 2.2(a). Such an architecture introduces bus contention in task scheduling problems[71]. In this direction, researchers have proposed excellent theoretical results, e.g., [43, 57, 70]. Popular objectives in the optimization

are such as the pin/wiring minimization (for the cost consideration) [43, 70] and the makespan minimization (for the performance consideration) [57]. A popular and practical communication architecture in recent years is the *multi-bus architecture*, which have buses of different bandwidths connected by bridges/switches to provide different quality-of-service degrees to processing elements of different performance levels or purposes, as shown in Figure 2.2(b). With the popularity of such an architecture for embedded-system designs, how to schedule task with performance guarantees and bus-cost minimization becomes a very important design issue. Because of the difficulty in task scheduling, existing research results are mainly based on heuristics or search-based solutions, e.g., [52, 61].

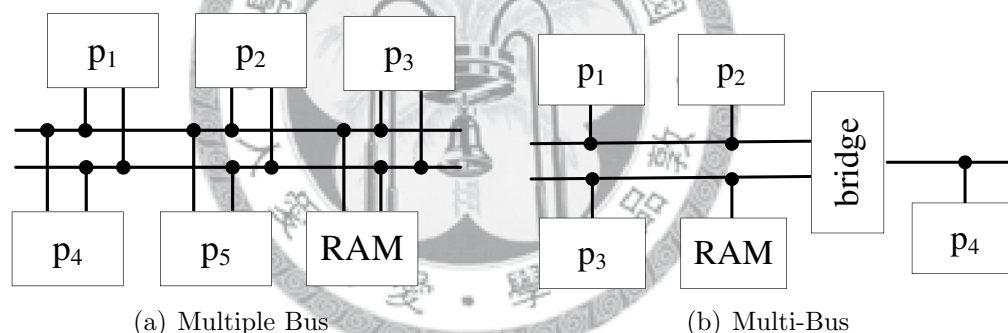


Figure 2.2: The multiple bus and multi-bus architectures

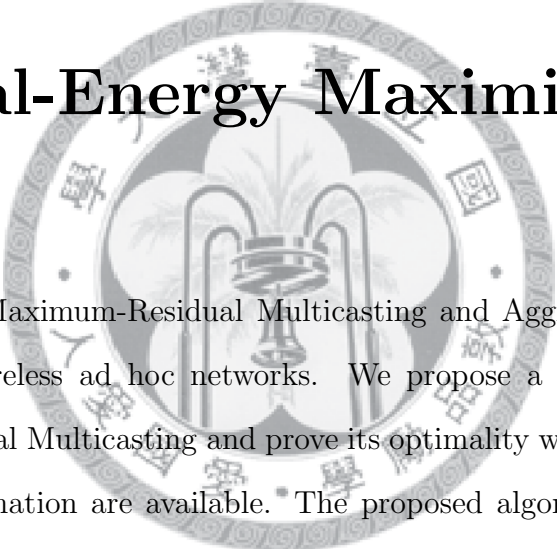
The *multi-layer bus architecture* provided by ARM [1] further improves the communication concurrency and flexibility [54], as shown in Figure 5.1. The multi-layer bus architecture is effectively utilized by many practitioners to fit the characteristics and needs of various embedded systems [93]. Example cases include MPEG decoding systems that adopt a high-speed bus to connect a digital signal processor and a memory controller and another peripheral bus to connect an analog-to-digital converter and serial ports, where traffics of different characteristics are separated [79]. In many of such embedded systems, tasks are often compiled to run on some specific (types of) processors and expected to execute with deadline constraints. Although excellent design methodologies and EDA tools have been

developed to address (hardware) architecture synthesis issues in processor allocation, *e.g.*, [9, 65], and bus partitioning, *e.g.*, [46, 68], little work has been done in the exploration of the trade-off between the cost and performance in system designs. The trade-off issue has become even more critical with the increasing number of bus layers because it implies the overheads of the cost per area, power consumption and design complexity.



Chapter 3

Residual-Energy Maximization



In this chapter, Maximum-Residual Multicasting and Aggregating are explored in heterogeneous wireless ad hoc networks. We propose a *Prim*-like algorithm for Maximum-Residual Multicasting and prove its optimality when up-to-date topology and energy information are available. The proposed algorithm can be applied to many existing routing protocols, especially those based on the link-state approach, *e.g.*, [28, 35]. In practical implementation, control messages for maintaining up-to-date topology and energy information need additional energy consumption. We show by experiments that the proposed algorithm for Maximum-Residual Multicasting intends to find the best route in a dynamic fashion to prolong network lifetime. The experimental results show that it is excellent in the improvements of network lifetime and load balance, in comparison with previous related work, *e.g.*, [50, 72, 77, 85]. On the other hand, we show that Maximum-Residual Aggregating is \mathcal{NP} -hard and approximation algorithms for this problem are unlikely to exist. We then prove that, unless $\mathcal{P} = \mathcal{NP}$, its minimization version cannot be approximated in polynomial time within a ratio of $(2 - \epsilon)$ for any $\epsilon > 0$, where the ratio bound is with respect

to the maximum remaining energy before routing.

The rest of this chapter is organized as follows: In Section 3.1, the network model under consideration is described. We then formulate the Maximum-Residual Multicasting and Aggregating problems in a more formal way. Sections 3.2 and 3.3 provide a positive and negative result for the two problems, respectively. Section 3.4 summarizes the experimental results and provides observations. Section 3.5 gives a brief summary.

3.1 Network Model and Problem Formulation

Maximum-Residual Routing is explored in a heterogeneous wireless ad hoc network with nodes deployed in a 3-dimensional area. Each node is equipped with a wireless transceiver. Transmission by a node can be received by all nodes that lie within its communication range (depending on its antenna pattern and transmission power levels). Reception of a node from different nodes cannot proceed together in one reception. The energy consumption model is similar to that in [33]. The energy consumed by a transmitter u is proportional to $\tau(u) + \alpha \times \delta(u, v)^\kappa$, where $\tau(u)$ is a constant based on the *transmit amplifier*, $\delta(u, v)$ is the *Euclidean distance* between nodes u and v , $\alpha \geq 1$ is the *transmission-quality parameter* depending on the antenna designs, and $\kappa \in [2, 4)$ is the *distance-power gradient* depending on the environment conditions. The energy consumed by the receiver v , on the other hand, is a constant $\gamma(v)$, related to the time and the energy that its transceiver spends and consumes in receive mode. Nodes may have different values of γ because wireless devices may be equipped with different transceivers produced by different vendors. Each node in the network could be either a source or a router. When having packets

to multicast to or aggregate from a set of destinations, the source needs to establish a multicast or an aggregate tree with the destinations for routing the packets. A multicast tree is a directed tree having one source without any *incoming* edge and all other nodes with exactly one. An aggregate tree is a directed tree having one source without any *outgoing* edge and all other nodes with exactly one. Figures 3.1(a) and 3.1(b) show an example network topology on which a multicast tree and an aggregate tree are determined, respectively.

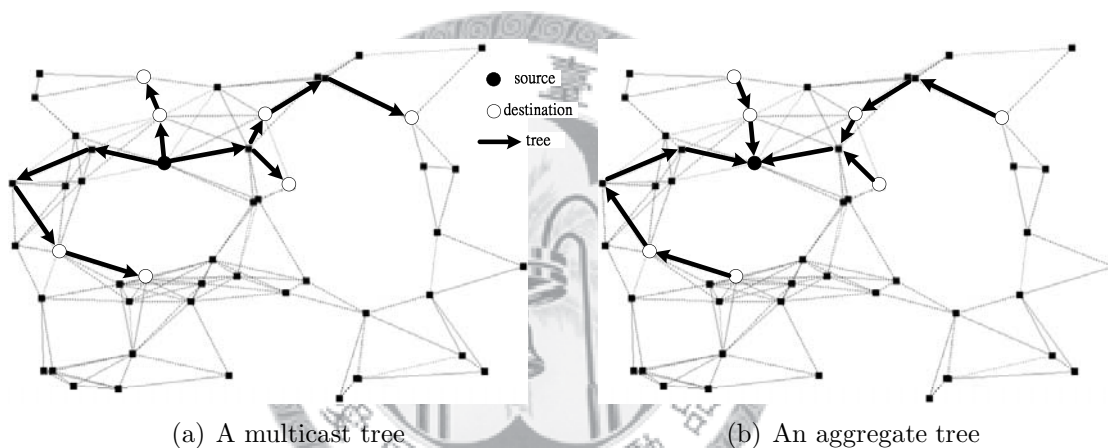


Figure 3.1: A network topology example

The problems under discussion can be defined as follows: A network topology $G = (V, E)$ is a directed graph in which V is the set of nodes, and E is the set of communication links between pairs of nodes. For each node $u \in V$, a budget $\beta(u)$ denotes the remaining energy of u before routing. When a source s has packets to multicast or aggregate, for each directed edge $(u, v) \in E$, let a weight $\omega(u, v)$ denote the energy consumption of u for transmitting the packets to v and a constant $\gamma(v)$ denote the energy consumption of v for receiving the packets. Because of the wireless medium, transmitting packets to multiple nodes lying in the communication range needs only one transmission, while receiving packets from multiple nodes needs to consume energy for each reception. Therefore, the remaining budget of u

(after routing on T) can be defined as

$$\hat{\beta}_T(u) = \beta(u) - \max_{(u,v) \in T} \omega(u,v) - \sum_{(w,u) \in T} \gamma(u),$$

where $(u,v) \in T$ and $(w,u) \in T$ are the outgoing and incoming edges of u on T , respectively. Note that each node, except for s , in a multicast tree T , has exactly one incoming edge and may have several outgoing edges. The case is exactly the reverse in an aggregate tree (refer to Figure 3.1). This is the reason why the asymptotic hardness of the two problems is so different. For the convenience of discussions, let $\beta(V)$ and $\hat{\beta}_T(V)$ denote $\min\{\beta(u) \mid \forall u \in V\}$ and $\min\{\hat{\beta}_T(u) \mid \forall u \in V\}$, respectively. In the following, we call a tree T rooted at a source a *maximum-residual multicast* (or *aggregate*) *tree* if T spans all of the vertices in a given destination set and $\hat{\beta}_T(V) \geq \hat{\beta}_{T'}(V)$, where T' is any multicast (or aggregate) tree rooted at the source that also spans all of the vertices in the destination set.

Maximum-Residual Multicasting (Aggregating) Problem

Input instance: A network topology $G = (V, E)$, where each vertex $u \in V$ has a budget $\beta(u)$ and a constant $\gamma(u)$, and each edge $(u, v) \in E$ has a weight $\omega(u, v)$. A source $s \in V$ and a destination set $R \subseteq V$.

Objective: A multicast (or an aggregate) tree T rooted at s that spans all vertices in R such that $\hat{\beta}_T(V)$ is maximized.

3.2 An Optimal Algorithm for Maximum-Residual Multicasting

This section is to present our algorithm for Maximum-Residual Multicasting. Given a network topology $G = (V, E)$ with a budget function β , a weight function ω , a reception function γ , a source $s \in V$, and a destination set $R \subseteq V$, an optimal algorithm shown in Algorithm 1, referred to as *MRMT*, is proposed to produce a maximum-residual multicast tree T from s to R . For convenience, we call a partition of V a *cut* $(U, V - U)$ and an edge (u, v) a *stingy edge* crossing the cut $(U, V - U)$ if $u \in U$ and $v \in V - U$ such that $\min\{\beta(u) - \omega(u, v) - \gamma(u), \beta(v) - \gamma(v)\}$ is maximized. An observation is that every spanned vertex, except for s , has to consume energy for its reception. For the simplification of discussions, we add $\gamma(s)$ to $\beta(s)$ in the beginning (Line 1) and subtract additional $\gamma(s)$ from $\beta(s)$ at the end (Line 9). It can be imaged that an auxiliary source s' with $\beta(s') = \infty$ and an edge (s', s) with $\omega(s', s) = 0$ are introduced and have been spanned. Then, MRMT starts from adding s into V_T , a set of vertices spanned by T (Line 3), and goes on spanning other vertices in $V - V_T$ until all the vertices in R are spanned (Line 4). At each step, a stingy edge crossing the cut $(V_T, V - V_T)$ is added to T (Lines 5-7). When all vertices in R are spanned, there are $|V_T|$ vertices spanned and exactly $|V_T| - 1$ edges added to T . Eventually, T forms a multicast tree. According to T , each vertex u in V_T is actually assigned the power level such that the packets can be multicasted from s to R , and has a remaining budget $\beta'_T(u) = \beta(u) - \max\{\omega(u, v) \mid \forall (u, v) \in E_T\} - \gamma(u)$ (Lines 8-9).

The time complexity of MRMT depends on the time required to find a stingy edge to grow T . A straightforward method finds such an edge by searching

Algorithm 1 MRMT: a polynomial-time optimal algorithm for Maximum-Residual Multicasting

Input: A network topology $G = (V, E)$ with a budget function β , a weight function ω , and a reception function γ , a source $s \in V$, and a destination set $R \subseteq V$

Output: A maximum-residual multicast tree $T = (V_T, E_T)$ from s to R

- 1: $\beta(s) \leftarrow \beta(s) + \gamma(s)$
 - 2: $E_T \leftarrow \phi$
 - 3: $V_T \leftarrow \{s\}$
 - 4: **while** $R \not\subseteq V_T$ **do**
 - 5: Find an edge (u, v) crossing the cut $(V_T, V - V_T)$ such that $\min\{\beta(u) - \omega(u, v) - \gamma(u), \beta(v) - \gamma(v)\}$ is maximized.
 - 6: $E_T \leftarrow E_T \cup \{(u, v)\}$
 - 7: $V_T \leftarrow V_T \cup \{v\}$
 - 8: **for all** $u \in V_T$ **do**
 - 9: $\hat{\beta}_T(u) \leftarrow \beta(u) - \max\{\omega(u, v) \mid \forall (u, v) \in E_T\} - \gamma(u)$
-

the adjacency lists of the vertices in V . Each step costs $O(E)$ time, and we conclude a total running time of $O(|E||V|)$. It is not hard to see that MRMT can be improved to run in $O(|E| + |V| \log |V|)$ time by using *Fibonacci heaps*, since MRMT bears similarity to *Prim's algorithm* for computing *minimum-spanning trees* [21], *i.e.*, a tree that spans all of the vertices in V with the minimum total weight of edges. The optimality of MRMT, however, is not so obvious as the time complexity that can directly be derived from Prim's algorithm. A variant of Minimum-Spanning Tree is referred to as Minimum-Steiner Tree, which is also required to span only a subset of vertices and is proved to be \mathcal{NP} -complete [29]. In the variant, a challenge is to determine which vertices should serve as the intermediates so as to form an optimal solution. It results in the \mathcal{NP} -completeness of Minimum-Steiner Tree and would be a challenge for Maximum-Residual Multicasting as well. The following theorem proves that MRMT is an optimal algorithm for Maximum-Residual Multicasting.

Theorem 3.1 *Algorithm MRMT always derives a maximum-residual multicast tree from s to R .*

Proof. The theorem is proved by showing an invariant that each step of MRMT always derives a subtree of some optimal solution. To be more specific, given a tree that is a subtree of any maximum-residual multicast tree from s to R and has not yet spanned all vertices in R , MRMT always includes a new edge that will not violate the invariant. When all vertices in R are spanned, a maximum-residual multicast tree is thus derived.

Let T be a subtree of some maximum-residual multicast tree T^* from s to R , V_T be the set of vertices that T spans, and (u, v) be a stingy edge crossing the cut $(V_T, V - V_T)$. If T has not yet spanned all of the vertices in R , we show that the inclusion of the stingy edge (u, v) to T will not result in the violation of the invariant. Suppose that T^* does not contain (u, v) (if it does, we are done). We show another maximum-residual multicast tree T' that includes $T \cup \{(u, v)\}$ can be constructed from T^* . We delineate two cases, depending on whether $v \in T^*$ or not:

Case 1: Suppose $v \in T^*$. We remove the incoming edge of v from T^* and then add the stingy edge (u, v) to T^* . It is clear that T' is a legal multicast tree that contains $T \cup \{(u, v)\}$ and spans all vertices in R . We now show that $\hat{\beta}_{T'}(V) \geq \hat{\beta}_{T^*}(V)$. The removing of an edge will not let any vertex decrease its budget, and the inclusion of the edge (u, v) only has a budget impact on u . Note that the budget of v does not change in this case. Because T^* spans all vertices in R , T^* must contain an edge (x, y) that crosses the cut $(V_T, V - V_T)$. Moreover, $\min\{\beta(x) - \omega(x, y) - \gamma(x), \beta(y) - \gamma(y)\} \leq \min\{\beta(u) - \omega(u, v) - \gamma(u), \beta(v) - \gamma(v)\}$, since (u, v) is a stingy edge that crosses the cut $(V_T, V - V_T)$. It means that $\beta(u) - \omega(u, v) - \gamma(u) \geq \min\{\beta(x) - \omega(x, y) - \gamma(x), \beta(y) - \gamma(y)\} \geq \hat{\beta}_{T^*}(V)$. Therefore, we conclude that $\hat{\beta}_{T'}(V) \geq \hat{\beta}_{T^*}(V)$.

Case 2: Suppose that $v \notin T^*$. We just add the stingy edge (u, v) to T^* . The

inclusion of the new edge (u, v) only affects the budgets of u and v . The correctness of $\beta(u) - \omega(u, v) - \gamma(u) \geq \hat{\beta}_{T^*}(V)$ follows directly from the analysis in Case 1. On the other hand, since $v \notin T^*$, the remaining budget of v on T' , *i.e.*, $\hat{\beta}_{T'}(v)$, is $\beta(v) - \gamma(v)$. There must be an edge $(x, y) \in T^*$ crossing the cut $(V_T, V - V_T)$ such that $\min\{\beta(u) - \omega(u, v) - \gamma(u), \beta(v) - \gamma(v)\} \geq \min\{\beta(x) - \omega(x, y) - \gamma(x), \beta(y) - \gamma(y)\}$. In other words, $\beta(v) - \gamma(v) \geq \hat{\beta}_{T^*}(V)$. Because T^* is a maximum-residual multicast tree, T' must be another maximum-residual multicast tree that contains $T \cup \{(u, v)\}$.

Throughout, MRMT always adds stingy edges to T . Since the initial tree $T = s$ must be a subtree of some maximum-residual multicast tree, and MRMT terminates when the derived subtree spans all vertices in the destination set, MRMT derives a maximum-residual multicast tree from s to R . ■

We must point out that the multicast tree derived by MRMT may have some redundant edges to prune without losing the spanning of all vertices in R . Redundant edges can be pruned by simply traversing the multicast tree to prune edges from the vertices if neither themselves nor their descendants are in R . It can be done in $O(|V|)$ time. Note that edge pruning does not result in the budget decreasing of any vertex, and the derived multicast tree after the above edge pruning remains a maximum-residual multicast tree.

3.3 A $(2 - \epsilon)$ -Inapproximability Result for Maximum-Residual Aggregating

Unlike Maximum-Residual Multicasting, Maximum-Residual Aggregating is unfortunately \mathcal{NP} -hard. In this section, we present the hardness of Maximum-Residual

Aggregating.

Lemma 3.1 *Maximum-Residual Aggregating is \mathcal{NP} -hard.*

Proof. We prove this lemma by a reduction from the decision version of *Hamiltonian Path*, which is known to be \mathcal{NP} -complete [29]. The input to the Hamiltonian Path problem is an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. The output is *YES* if and only if \mathcal{G} has a simple path that contains every vertex in \mathcal{V} . In order to distinguish directed edges from undirected edges, let us denote the undirected edge incident on u and v in \mathcal{G} by $[u, v]$.

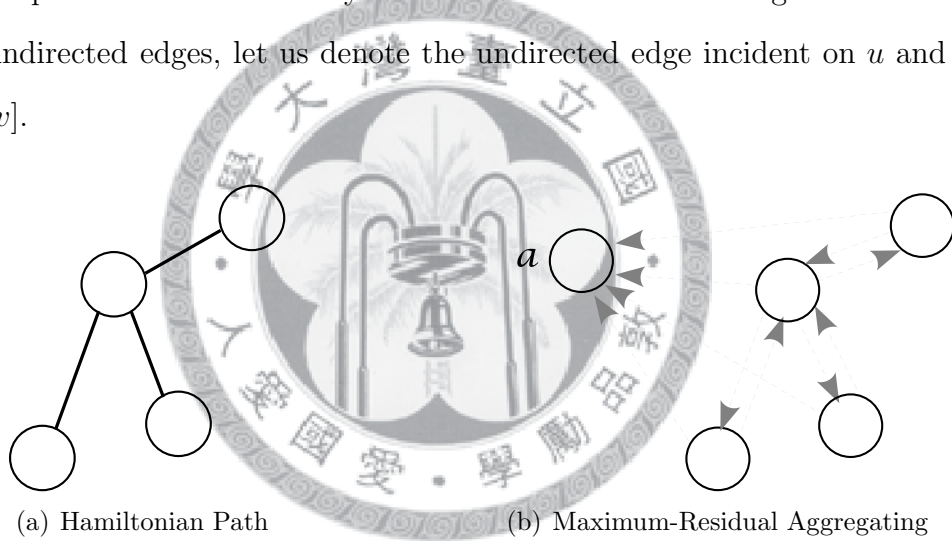


Figure 3.2: An input instance example

For any given instance $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ of Hamiltonian Path, an example shown as Figure 3.2(a), we show how to construct, in polynomial time, an instance $\langle G = (V, E), \beta, \omega, s, R, \gamma \rangle$ of Maximum-Residual Aggregating such that \mathcal{G} has a Hamiltonian path if and only if G contains an aggregate tree T with $\hat{\beta}_T(V) = k$, where k is a non-negative constant. The construction is as follows: First, we add a new vertex $a \notin \mathcal{V}$. Let $V = \mathcal{V} \cup \{a\}$ and $E = \{(u, v), (v, u) \mid \forall [u, v] \in \mathcal{E}\} \cup \{(u, a) \mid \forall u \in \mathcal{V}\}$, as shown in Figure 3.2(b). Then, we assign the corresponding values to each vertex and edge. For each $(u, v) \in E$, let $\omega(u, v) = w$, where $w \geq 0$. For each $u \in V$, let $\gamma(u) = r$, where $r > 0$. Let $\beta(a) = \mathcal{B} - w$ and $\beta(u) = \mathcal{B}$ for each $u \in V - \{a\}$, where

$\mathcal{B} = w + |\mathcal{V}| \times r$. Let $s = a$ and $R = V - \{s\}$. The instance can be constructed in a polynomial time of $|\mathcal{V}|$ and $|\mathcal{E}|$.

To complete the proof, we now show that \mathcal{G} has a Hamiltonian path if and only if G contains an aggregate tree T with $\hat{\beta}_T(V) = \mathcal{B} - w - r$. Suppose that \mathcal{G} has a Hamiltonian path. Since (u, v) and $(v, u) \in G$ if $[u, v] \in \mathcal{G}$, G must contain an aggregate tree T in which every vertex has at most one outgoing edge and at most one incoming edge. Thus, $\hat{\beta}_T(V) = \mathcal{B} - w - r$. On the other hand, suppose that \mathcal{G} has no Hamiltonian path. Any aggregate tree T of G must contain a vertex with at least two incoming edges, and thus $\hat{\beta}_T(V) \leq \mathcal{B} - w - 2r$. Since this construction can be done in polynomial time, the existence of a polynomial-time algorithm for Maximum-Residual Aggregating implies the same for Hamiltonian Path. We conclude that Maximum-Residual Aggregating is \mathcal{NP} -hard. ■

Since Maximum-Residual Aggregating is \mathcal{NP} -hard, there is no polynomial-time optimal algorithm, unless $\mathcal{P} = \mathcal{NP}$. Moreover, approximation algorithms for this problem are equally unlikely to exist. Indeed, since the optimum can be arbitrary close to zero, the approximation ratio of any polynomial-time algorithm becomes arbitrarily large. A simple observation allows us to sidestep this problem and provide a meaningful baseline for approximation algorithms: The optimal solution to maximize the minimum remaining budget, *i.e.*, $\hat{\beta}_T(V)$, is equivalent to the optimal solution to minimize the maximum budget utilization, *i.e.*, $\mathcal{B} - \hat{\beta}_T(V)$, for some sufficiently large constant \mathcal{B} . We are interested in approximation algorithms for the minimization of the maximum budget utilization with respect to the maximum remaining budget before routing, *i.e.*, $\mathcal{B} = \max\{\beta(u) \mid u \in V\}$, and refer to the problem as the minimization version of Maximum-Residual Aggregating. Note that an α -approximation algorithm for the minimization version does not translate into

an α -approximation algorithm for Maximum-Residual Aggregating (and vice versa).

Theorem 3.2 *The minimization version of Maximum-Residual Aggregating cannot be approximated in polynomial time within a ratio of $(2 - \epsilon)$ for any $\epsilon > 0$, with respect to the maximum remaining budget before routing \mathcal{B} , unless $\mathcal{P} = \mathcal{NP}$.*

Proof. The proof follows from Lemma 3.1. By multiplying $\hat{\beta}_T(V)$ with -1 and the adding of \mathcal{B} , the reduction in Lemma 3.1 can be rephrased as follows: \mathcal{G} has a Hamiltonian path if and only if \mathcal{G} contains an aggregate tree T with $\mathcal{B} - \hat{\beta}_T(V) = w + r$.

This theorem can be proved by contradiction. Suppose that there were a polynomial-time ρ -approximation algorithm \mathcal{A} for the problem, for some approximation ratio $\rho < 2$. We show how to use the hypothetical algorithm \mathcal{A} to decide whether \mathcal{G} has a Hamiltonian path. Because \mathcal{A} is an approximation algorithm with a ratio ρ , \mathcal{A} will output an aggregate tree T with $\mathcal{B} - \hat{\beta}_T(V) \leq \rho(w + r)$ if \mathcal{G} has a Hamiltonian path; otherwise, \mathcal{A} will output T with $\mathcal{B} - \hat{\beta}_T(V) \geq w + 2r$. It implies that \mathcal{A} can be used to decide whether \mathcal{G} has a Hamiltonian path if $\rho < \frac{w+2r}{w+r}$. Hence, unless $\mathcal{P} = \mathcal{NP}$, no polynomial-time algorithm can be guaranteed to derive an aggregate tree T with $\mathcal{B} - \hat{\beta}_T(V) \leq (\frac{w+2r}{w+r} - \epsilon) \times (\mathcal{B} - \hat{\beta}_{T^*}(V))$, for any $\epsilon > 0$ and any optimal solution T^* . The ratio bound approaches $(2 - \epsilon)$, as $\frac{w}{r}$ approaches 0. ■

Note that the negative results hold for general network topologies. In some specific applications, nodes are applied in suburban areas with omnidirectional antenna patterns. In that case, a network topology is obtained by considering circular communication ranges assigned to nodes in the 3-dimensional Euclidean space. It remains open whether this geometric special case can be solved in polynomial time, despite the NP-hardness of its general graph version.

3.4 Performance Evaluation

In Section 3.2, an optimal algorithm is proposed for Maximum-Residual Multicasting. The problem explored in this work needs the information of the network topology and the remaining energy of nodes up to a certain precision. However, the network topology and the remaining energy information of nodes may change with time. In practice, the algorithm can be applied to use in existing link-state routing protocols with similar implementations/designs, *e.g.*, STAR [28] and OLSR [35], where network topology information is collected and maintained in an up-to-date way by broadcasting the link-state costs of adjacent nodes to other nodes [5]. The broadcasting of energy information can be done in a similar way, except that we have to define a notification threshold on the remaining energy changing of nodes. The smaller the threshold, the more precise the energy information but the heavier the control traffic. We explore the impacts of the notification threshold values on the performance of the algorithm in the experiments.

3.4.1 Algorithms for Comparison

We evaluated the capability of the proposed algorithm MRMT in the prolongation of network lifetime, in comparison with four multicast algorithms [50, 72, 77, 85] presented as follows:

- The maximization of the minimum remaining energy of nodes *before* routing [50, 77], *i.e.*, the maximization of $\min_{u \in T} \beta(u)$, is to avoid routing packets over nodes that have low remaining energy, where T denoted the derived routing tree. The *Bellman-Ford* algorithm was revised for the derivation of routing

trees for the performance evaluation (referred to as *MMENT*).

- The minimization of the maximum energy consumption of nodes for routing packets [72], *i.e.*, the minimization of $\max_{u \in T} \{\beta(u) - \hat{\beta}_T(u)\}$, is to avoid the consumption of too much energy at a single node for routing the packets. *Prim's algorithm* was revised in performance evaluation (referred to as *MSMT*).
- While Minimum-Energy Multicasting has proved to be \mathcal{NP} -hard [7, 12, 20], excellent heuristics were proposed to minimize the total energy consumption in the network for multicasting, *i.e.*, the minimization of $\sum_{u \in T} (\beta(u) - \hat{\beta}_T(u))$. The heuristic algorithm *BIP*, proposed in [85], was adopted in performance evaluation (referred to as *BIPMT*).
- Another algorithm for comparison is on the minimization of the maximum energy consumption of any path [85], *i.e.*, the minimization of $\max_{v \in R} \{\sum_{u \in s \rightsquigarrow v} (\beta(u) - \hat{\beta}_T(u))\}$, where $s \rightsquigarrow v$ is the path from s to v in T . It is to minimize the total energy consumed in forwarding packets to one of the destinations. *Dijkstra's algorithm* was adopted in the derivation of a routing tree (referred to as *SPMT*).

3.4.2 Experimental Setups and Performance Metrics

The experimental environment consisted of a specified number of nodes ($|V| = 10, 20, \dots, 100$). Nodes were randomly placed in a 3-dimensional rectangular area ($100 \times 100 \times 20$). Each node was equipped with an omnidirectional antenna (transmission range 0-40) and a power supplier (battery capacity 1×10^8). When having packets to multicast to a set of destinations, a source had to establish a session with

the destinations. In the experiments, we assume that the number of packets routed in a session was fixed and equal to 100 (but the algorithms can be used without this restriction). We adopted an energy consumption model similar to the model in [33]. The energy consumed for transmitting all the packets of a session within distance δ was set as $1\% \times 40^3 + 1 \times \delta^3$, where $1\% \times 40^3$, *i.e.*, 1% of the energy consumption for the transmitter at the maximum transmission power, was consumed by the *transmit amplifier*. The impacts of different energy values consumed in receiving all the packets of a session was also reported in the experiments ($\gamma = (1\%, 2\%, \dots, 10\%) \times 40^3$). For a network, one node was randomly chosen as the source every time, and the source had packets that need to establish 1-100 sessions to multicast to a destination set (randomly chosen from the rest). The *uniform distribution* was adopted for the random choices in the experiments. This process was repeat until any node had its energy exhausted.

In addition to data packets, extra control messages were needed to advertise energy information when MRMT and MMEMT were adopted as routing algorithms, since MRMT and MMEMT considered the remaining energy of nodes to derive routing trees (while the other three did not). The control messages allowed all nodes in the network to have an identical view so that all nodes can determine consistent routes in a distributed way. A threshold Δ was set for the notification of the change in the remaining energy of a node. Whenever the remaining energy of a node decreased by an amount equal to Δ , the node broadcasted a control message (with size equal to that of a data packet) to all reachable nodes so as to update the energy information. The impacts of different notification thresholds were also explored ($\Delta = (5, 10, \dots, 50) \times 10^3$).

For the fairness in performance comparison, every routing algorithm was

evaluated over the same data sets. In the study, two performance metrics were adopted: (1) *Network Lifetime* (measured in terms of the average number of sessions a node can establish) and (2) *Load Balance* (measured in terms of the standard deviation of the remaining energy of nodes). Other performance metrics adopted generally for routing protocol evaluation, *e.g.*, propagation delay and delivery ratio, highly depend on the routing protocol which the routing algorithms are incorporated with, and are not studied here. For each routing algorithm, the experimental results were derived as an average value of those of 100 independent experiments.

3.4.3 Experimental Results

Figures 3.3(a) and 3.3(b) show the network lifetime and the load balance impacted by the network size ($|V| = 10, 20, \dots, 100$) under $\gamma = 1\% \times 40^3$ and $\Delta = 10 \times 10^3$. As shown in Figure 3.3(a), the network lifetime increases, in general, as the number of nodes increases. The main reason behind the observation is that a smaller network size in the same 3-dimensional space may result in fewer alternatives in routing packets and earlier draining away of energy of some nodes. As the network size increases in the same 3-dimensional space, a higher node density of nodes may result in more available paths between nodes, and more energy-efficient routes could be determined. When $|V| \geq 50$, MRMT could improve the network lifetime by 48% to 220%, compared with different other algorithms. Because of a similar reason, the standard deviation of the remaining energy of nodes decreases as the number of nodes increases, as shown in Figure 3.3(b). When $|V| \geq 50$, MRMT could reduce the variance by 39% to 98%, compared with different other algorithms. These observations imply that the attempt in keeping the minimum remaining energy as high as possible has a more balance degree in the remaining energy of nodes and a

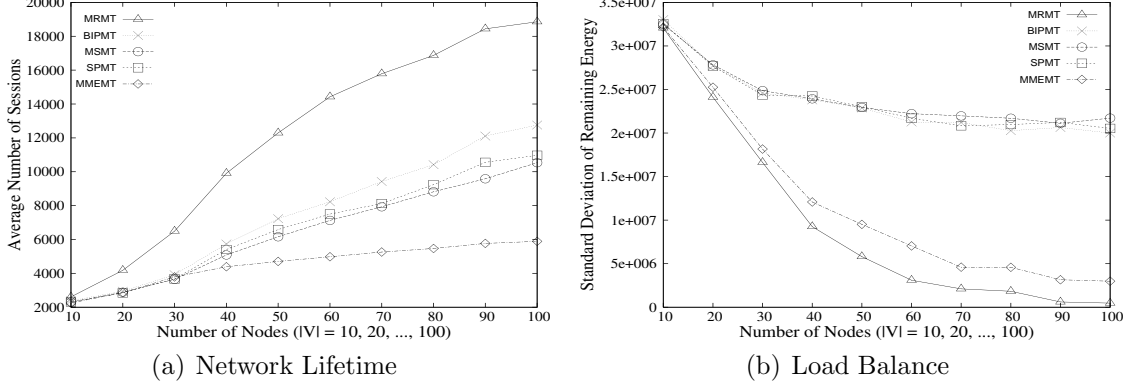


Figure 3.3: Impacts of the network size ($\gamma = 1\% \times 40^3$, $\Delta = 10 \times 10^3$)

longer network lifetime. The attempt also tends to do an even better job when a network becomes more dense.

Figures 3.4(a) and 3.4(b) show the network lifetime and the load balance of different algorithms when different amounts of energy consumption for receiving packets were considered, *i.e.*, $\gamma = (1\%, 2\%, \dots, 10\%) \times 40^3$ with $|V| = 50$ and $\Delta = 10 \times 10^3$. As shown in Figure 3.4(a), the network lifetime decreases when the energy consumption for reception increases. Because more energy was consumed for each reception, we observed quicker exhaustion of node energy. Nevertheless, it is worth mentioning that MRMT achieves different degrees of improvement with different γ values. When $\gamma = 1\% \times 40^3$ and $10\% \times 40^3$, MRMT could improve the network lifetime by 70% to 161% and 29% to 76%, respectively, compared with different other algorithms. We have to point out that the setup value of the notification threshold (*i.e.*, $\Delta = 10 \times 10^3$) may play a big role in these experiments. An improper setup of the notification threshold might result in heavy energy consumption for control traffic, and the exhaustion of node energy would be accelerated (see the following experiments on the impacts of notification thresholds). As shown in Figure 3.4(b), the amount of energy consumption for reception has an insignificant impact on the load balance issue. MRMT and MMEMT can significantly reduce the variance in

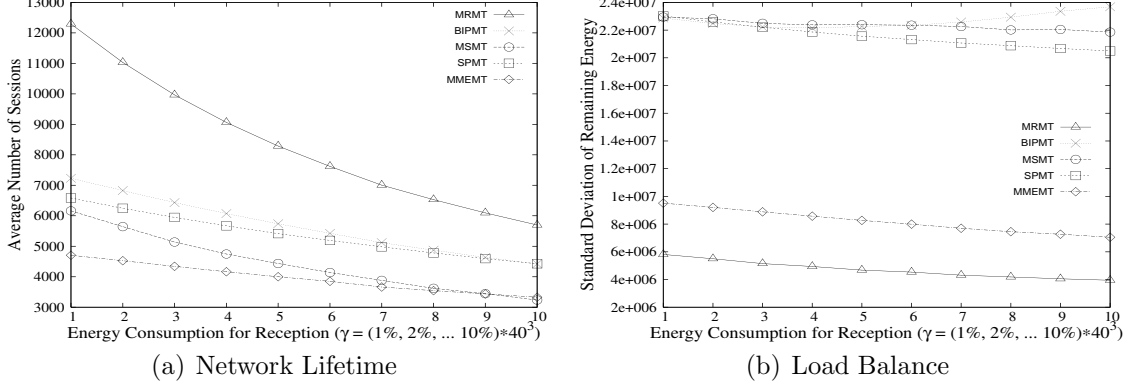


Figure 3.4: Impacts of the energy consumption for reception ($|V| = 50$, $\Delta = 10 \times 10^3$)

the remaining energy of nodes (with better load balancing), compared with other algorithms. It is because MRMT and MMEMT consider the remaining energy of nodes when determining routes, and they attempt to route packets in a load-balancing way. Moreover, MRMT further outperforms MMEMT in the load balance since MRMT considers the energy consumption for transmissions (*i.e.*, the weights of edges in a network topology), but MMEMT does not.

Figures 3.5(a) and 3.5(b) show the network lifetime and the balance of remaining energy of nodes with respect to the notification thresholds, *i.e.*, $\Delta = (5, 10, \dots, 50) \times 10^3$ with $|V| = 50$ and $\gamma = 1\% \times 40^3$. As shown in Figure 3.5(a), MRMT has the best performance improvement when $\Delta = 10 \times 10^3$. The reason for the improvement hiking from $\Delta = 5 \times 10^3$ to $\Delta = 10 \times 10^3$ is that the notification threshold was unduly small under the network setting when $\Delta = 5 \times 10^3$, and it may result in frequent dissemination of control messages. We must point out that the larger value the notification threshold is, the more out-of-date remaining energy information is used in determining routes. That is why the network lifetime decreases as the notification threshold increases when $\Delta \geq 10 \times 10^3$. With the same reason, out-of-date information may result in improper routing of packets over nodes with lower remaining energy. That also results in the increasing of the variance of

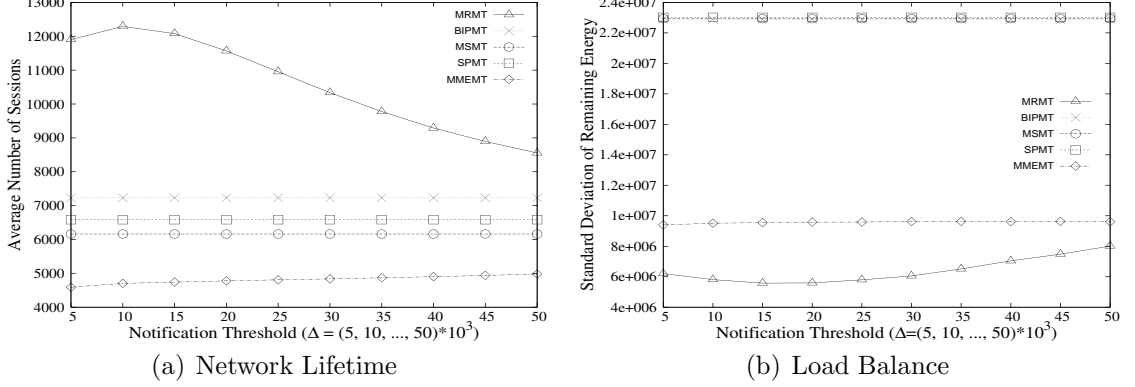


Figure 3.5: Impacts of notification thresholds ($|V| = 50$, $\gamma = 1\% \times 40^3$)

the remaining energy of nodes. As shown in the experiments, the setup of the notification threshold has no impacts on BIPMT, MSMT, and SPMT because they do not consider the remaining energy of nodes and have no control messages for such information collection.

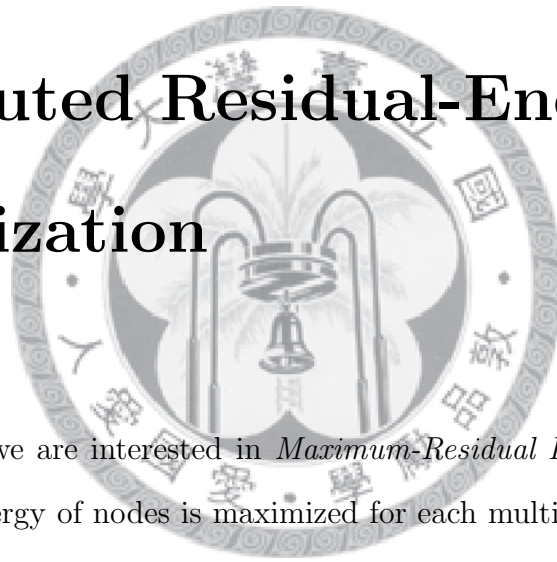
The experimental results show the importance in the design of protocols in the information maintenance of the remaining energy of nodes. The network lifetime and the balance in the remaining energy of nodes can be thus significantly improved. Another interesting observation is: The consideration of energy consumption in packets transmissions is more influential in the prolongation of network lifetime, while the consideration of remaining energy of nodes is more influential in the improvement of load balance. Based on the experimental results, we surmise that Maximum-Residual Routing would also have excellent performance when the delay in the partitioning of the network is considered (instead of the delay in the first node failure time). It is because the partitioning of the network results from a series of node failures, and this routing metric tends to prolong the node failure time.

3.5 Summary

This chapter targets power-aware routing in heterogeneous wireless ad-hoc networks. The routing metric to maximize the minimum remaining energy of nodes *after* packets of a session are routed is referred to as Maximum-Residual Routing. The objective is to keep the minimum remaining energy of all nodes as high as possible so as to delay the first failure time of nodes in the network. In this paper, we propose an algorithm for Maximum-Residual Multicasting and prove its optimality when up-to-date topology and energy information are available. The proposed algorithm is easy to implement and can be applied to use in existing link-state routing protocols for wireless ad-hoc networks, especially for those considering shortest-path routing by using Dijkstra's algorithm. We prove that Maximum-Residual Aggregating is \mathcal{NP} -hard and that, unless $\mathcal{P} = \mathcal{NP}$, its minimization version cannot be approximated in polynomial time within a ratio of $(2 - \epsilon)$ for any $\epsilon > 0$, with respect to the maximum remaining energy before routing. We have conducted extensive simulations to better understand the properties of the routing metric. Based on the experiments, we provide some interesting observations and conclude that using our routing metric to find routes is very beneficial because (1) network lifetime is significantly improved and (2) variance in remaining energy is significantly reduced, compared with other work [50, 72, 77, 85].

Chapter 4

Distributed Residual-Energy Maximization



In this chapter, we are interested in *Maximum-Residual Routing*, where the minimum residual energy of nodes is maximized for each multicast. The objective is to prolong the first node failure time when network topologies and data traffic may change frequently in an unpredictable way. This concept is first raised by Singh, et al. [72, 73], where no algorithm design and protocol implementation are presented in the work. Other closely related results are a heuristic algorithm for unicasting [88] and an optimal algorithm for broadcasting [56]. They are essentially algorithms relying on the knowledge of the entire topology and the remaining energy information of all nodes. Unlike the past work, we consider applications with a huge population of mobile devices such that no global information can be efficiently maintained at any node. We first propose a distributed algorithm for *Maximum-Residual Multicast* and prove its optimality without the considerations of node movements and control overheads. When mobility and control message collisions are taken into considera-

tion, it is shown that every derived route remains loop-free and converges toward an optimal solution in the maximization of the minimum residual energy. Based on the proposed algorithm, we then develop a source-initiated on-demand routing protocol, referred to as *Maximum-Residual Multicast Protocol* (MRMP), which is adaptable to network topologies and resources that may change over time. In MRMP, no periodic control message is employed to collect routing information a priori or repair link breakages. Neither group membership nor neighbor relationship is maintained at a node by explicit control messages. When desiring a route, a source invokes a route-discovery procedure over the network, and the individual decisions of intermediate nodes form a loop-free multicast tree naturally. For the performance evaluation, the protocol was implemented over NS2 [26], and simulations were conducted extensively with parameters set based on a realistic commercial wireless device [2]. We have very encouraging results in essential performance metrics adopted generally for routing protocol evaluation [22].

The rest of this chapter is organized as follows: Section 4.1 provides formal formulation of the problem. In Section 4.2, a distributed algorithm is proposed, and its essential properties are proved. The routing protocol and design issues are then addressed in Section 4.3. Simulation results and analysis are reported in Section 4.4. Section 4.5 is the conclusion.

4.1 Network Model and Problem Formulation

Depending on the duration of each multicast request and the network mobility degree, each multicast request might need to be partitioned into multiple sessions. The problem is formulated as the maximization of the minimum remaining energy

of nodes in the network after each multicast session, where the remaining energy of a node after each multicast is referred to as its *residual energy* for the rest of this chapter. The goal is to derive a route so as to maximize the minimum residual energy (of nodes in the network) without collecting and storing the detailed topology and the remaining energy information of the whole network at any node. Note that a route derived for a session could be considered *valid* within some time interval, because it does not rely on any information collected a priori by periodic control messages.

The network model under considerations can be formulated as a directed graph $G = (V, E)$, where each node $u \in V$ is associated with its remaining amount of energy, denoted as $\beta(u)$. We consider directed edges between nodes because of the possibility for different nodes in consuming different energy in packet transmissions (with different communication ranges). Each directed edge $(u, v) \in E$ is associated with a weight $\omega(u, v)$ to denote the amount of energy needed for a node u to transmit one session of data packets to another node v , and a constant $\gamma(v)$ denotes the energy consumption of receiving one session of data packets for node v . Note that the transmission of a node by wireless medium can be received by all of the nodes within its communication range. With such a consideration, the *residual energy* of a node u over a multicast tree T can be defined as follows:

$$\hat{\beta}_T(u) = \begin{cases} \beta(u) - \max\{\omega(u, v) \mid \forall (u, v) \in T\} & \text{if } u \text{ is the source node,} \\ \beta(u) - \max\{\omega(u, v) \mid \forall (u, v) \in T\} - \gamma(u) & \text{otherwise.} \end{cases}$$

The above formula implies that each node in T , except the leaves, transmits exactly one session of data packets. Each node in T , except the source, receives exactly one session of data packets. A node is the *source* if it initiates the multicast of the data packets. A node is a *leaf* in a multicast tree if it receives data packets but does not

send them to others.

Given an ad hoc network G , we choose to find a proper multicast tree T to deliver data packets from a source to a set of destinations (reachable from s) so as to maximize the minimum residual energy of all of the nodes in G , *i.e.*, the maximization of $\min\{\hat{\beta}_T(u) \mid \forall u \in G\}$. Note that the residual energy of the nodes not in T remains unchanged in this multicast, *i.e.*, $\forall u \notin T, \hat{\beta}_T(u) = \beta(u)$. To simplify the presentation, we can consider only those nodes in T (instead of in G). We shall prove later that an optimal solution that maximizes the minimum residual energy of nodes in T can also maximize the minimum residual energy of all the nodes in G . The minimum residual energy of nodes in T is referred to as the *residual energy over T* and denoted as $\hat{\beta}(T) = \min\{\hat{\beta}_T(u) \mid \forall u \in T\}$. The problem is formally defined as follows:

The Maximum-Residual Multicast Problem:

Suppose that there is an ad hoc network $G = (V, E)$, where each node $u \in V$ is associated with an amount $\beta(u)$ of its remaining energy and a constant amount $\gamma(u)$ of energy in receiving one session of data packets, and each directed edge $(u, v) \in E$ is associated with a weight $\omega(u, v)$ to transmit one session of data packets from u to v . Given a source $s \in V$ and a destination set $R \subseteq V$, the problem is to find a multicast tree $T \subseteq G$ that is rooted at s and includes all nodes in R such that $\hat{\beta}(T)$ is maximized. The tree is referred to as a *maximum-residual multicast tree*.

4.2 A Distributed Optimal Algorithm for Maximum-Residual Multicasting

In this section, a distributed routing algorithm is proposed to resolve the maximum-residual multicast problem, and its essential properties, especially optimality, are proved.

4.2.1 Algorithm Description

We propose to revise the Chandy-Misra algorithm [76], which is a well-known distributed version of the Bellman-Ford algorithm and is originally designed to derive shortest-path trees [8, 21]. Based on each multicast tree T derived by the to-be-proposed algorithm, every node is able to adjust its power level in packet transmissions so that the residual energy over a network $G = (V, E)$ is maximized for a given multicast session \mathcal{S} . Let the source and the destination set of \mathcal{S} be denoted as s and R , respectively. Given each node $v \in V$ under considerations, $\pi[v]$ and $m[v]$ are used to keep track of its predecessor and an estimation on the residual energy over a path from s to itself during the execution of the algorithm, respectively.

Algorithm 2 MRMA

Procedure source s

- 1: **if** s has a session \mathcal{S} of data packets to multicast to nodes in R **then**
 - 2: Create an entry indexed by (s, \mathcal{S}) at s ;
 - 3: $m[s] \leftarrow \beta(s)$;
 - 4: $\pi[s] \leftarrow NIL$;
 - 5: Broadcast $msg\langle s, \mathcal{S}, \beta(s), m[s], 0 \rangle$ to all of its neighbors
-

The proposed algorithm is referred to as the *Maximum-Residual Multicast Algorithm* (MRMA) and is shown in Algorithm 3: When a session \mathcal{S} is initiated,

MRMA is invoked (Line 1). An entry associated with \mathcal{S} is created at s by setting $m[s]$ and $\pi[s]$ as $\beta(s)$ and NIL (Lines 2-4), respectively. Because s is the source, the residual energy over the path of only s remains as $\beta(s)$ (*i.e.*, its remaining amount of energy). A control message is then broadcasted to all of the neighbors of s (Line 5). For each node u , its control message carries the source identification s , the session number \mathcal{S} , its remaining energy $\beta(u)$, an estimation $m[u]$ on the residual energy over a path from s to itself, and the energy consumption $\gamma(u)$ for u in receiving one session. Note that s does not need to consume energy for the reception, and thus the last field in its control messages is set as 0.

Procedure a node v other than s

```

6: if  $v$  receives  $msg\langle s, \mathcal{S}, \beta(u), m[u], \gamma(u) \rangle$  from a neighbor  $u$  then
7:   if no entry is indexed by  $(s, \mathcal{S})$  at  $v$  then
8:     Create an entry indexed by  $(s, \mathcal{S})$  at  $v$ ;
9:      $m[v] \leftarrow 0$ ;
10:     $\pi[v] \leftarrow NIL$ ;
11:   if  $m[v] < \min\{m[u], \beta(u) - \omega(u, v) - \gamma(u), \beta(v) - \gamma(v)\}$  then
12:      $m[v] \leftarrow \min\{m[u], \beta(u) - \omega(u, v) - \gamma(u), \beta(v) - \gamma(v)\}$ 
13:      $\pi[v] \leftarrow u$ 
14:   Broadcast  $msg\langle s, \mathcal{S}, \beta(v), m[v], \gamma(v) \rangle$  to all of its neighbors

```

When a node v receives a control message from a neighbor u (Line 6), it should first check up whether an entry exists for the corresponding session. If not, an entry is created by setting the initial values of $m[v]$ and $\pi[v]$ as 0 and NIL , respectively (Lines 7-10). Lines 11-14 will result in the update of $m[v]$ and $\pi[v]$ and the broadcast of a control message to all of its neighbors if a path from s to v (through u) with higher residual energy is found: u is a better predecessor of v in terms of residual energy maximization if the current value of $m[v]$ is smaller than $\min\{m[u], \beta(u) - \omega(u, v) - \gamma(u), \beta(v) - \gamma(v)\}$, where $m[u]$ is the estimate on the residual energy over the current path from s to u in T , $(\beta(u) - \omega(u, v) - \gamma(u))$ denotes the residual energy of u over a path through the edge (u, v) , and $(\beta(v) - \gamma(v))$ denotes the residual energy of v . Note that v has all of the information needed in

making the decision, except the amount of energy consumed by u to transmit the packets of this session to v , *i.e.*, $\omega(u, v)$. This information cannot be carried in the control message, because u has no knowledge about this information. We will explain later how v can derive this information in the protocol design.

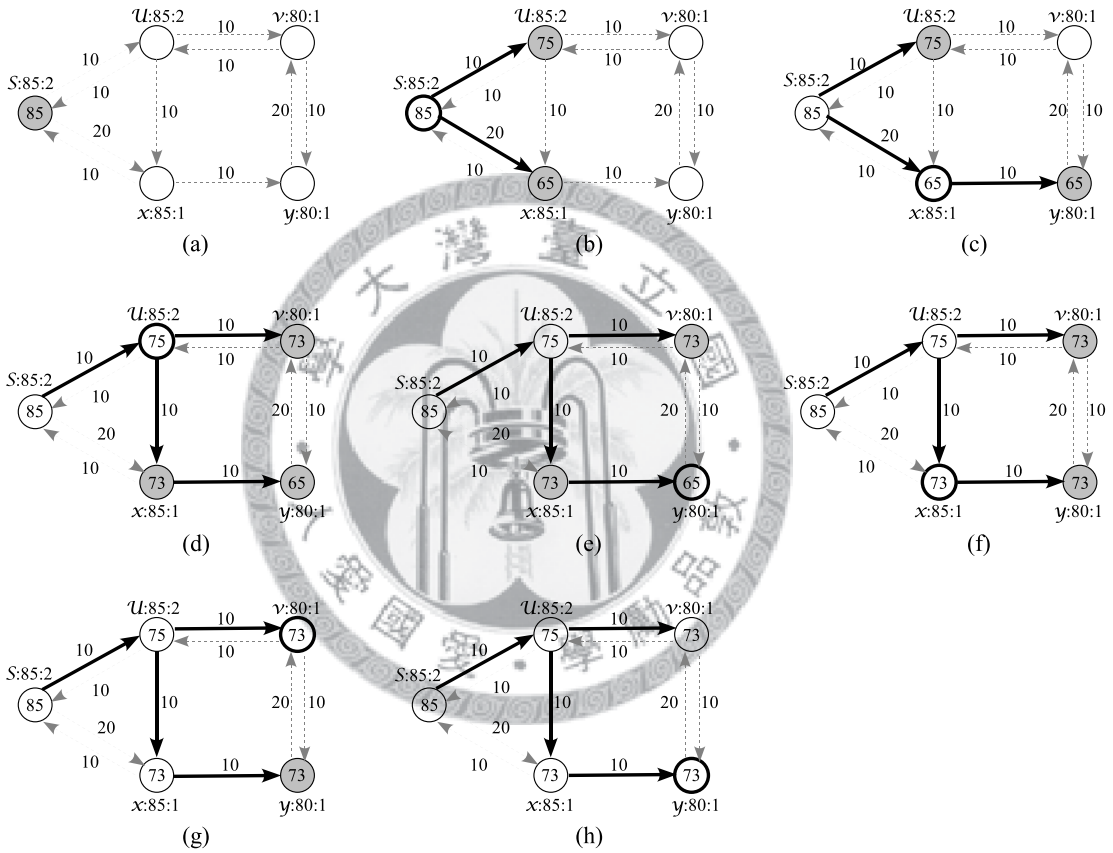


Figure 4.1: The execution of MRMA on an example network

MRMA is better illustrated by an example, as shown in Figure 4.1. The two values following each node symbol are the amounts of its remaining energy and its energy consumption of receiving one session (originated from s), *e.g.*, “ $s : \beta(s) : \gamma(s)$ ”. Suppose that all of the other nodes are the destinations of the session under discussion. The dotted edges indicate the possible links between pairs of nodes. Note that asymmetric links may exist because of directed edges. The weight lying on each direct edge denotes the amount of energy consumed by the transmitter to

transmit the session to the receiver. Each solid edge between two nodes indicates the predecessor relationship of the two nodes, and the value inside each node denotes the estimate on the residual energy over the current path from s to itself. Any node shown in the gray color indicates that there are control messages queued in the node for broadcasting. The node marked with a bold circle is a node which broadcasts a control message at that time moment.

In Figure 4.1(a), source s invokes MRMA by setting its estimate as the amount of its remaining energy 85 and broadcasting a control message. When nodes u and x receive the message from s , their estimates $m[u]$ and $m[x]$ are set as 75 and 65, respectively, as shown in Figure 4.1(b). Figure 4.1(c) shows the estimate $m[y]$ of node y after it receives the control message from x . Note that the link between x and u is asymmetric, and the control message from x is thus not received by u . Figure 4.1(d) shows the result after node u broadcasts its control message. Note that the estimate $m[x]$ of x increases to 73, and its predecessor $\pi[x]$ is reassigned to u . In Figure 4.1(e), node y broadcasts its control message, but nothing occurs. Because of the reassignment of $m[x]$ and $\pi[x]$ in Figure 4.1(d), node x broadcasts a control message to its neighbors again, as shown in Figure 4.1(f). As a result, $m[y]$ increases to 73 while $\pi[y]$ remains the same. Such a reassignment also causes node y to broadcast a control message again. Finally, as shown in Figures 4.1(f) and (h), no further changes occur after v and y broadcast their control messages. As we can see in Figure 4.1(h), a multicast tree can be derived by the predecessor relationship among nodes. Such a relationship can be used to let a predecessor node know the proper power level to send data packets to its successors. We shall address the remaining technical issues in a later section for a routing protocol design.

4.2.2 Properties

For the rest of this section, we shall show some essential properties of the proposed algorithm, especially the optimality of a derived route as a maximum-residual multicast tree. Note that the control overhead is highly dependent on the protocol design and is not considered here, where the assessment of the control overhead is done in the performance evaluation. Before further discussions, some terminology is defined: The *residual energy over a path p* is the minimum residual energy of the nodes on p (if the packets are routed through p), *i.e.*, $\hat{\beta}(p) = \min\{\hat{\beta}_p(u) \mid \forall u \in p\}$. A path p^* from s to v in G is called a *maximum-residual path* if the residual energy over p^* is no less than that over any path p from s to v in G , *i.e.*, $\hat{\beta}(p^*) \geq \hat{\beta}(p)$. The residual energy over a maximum-residual path from s to v is then denoted as $\mu(s, v)$. A *route T* derived at any time instance t by MRMA is defined as a subgraph (of the given network G) that is composed of the nodes already visited by control messages at t and the set of directed edges for the predecessor relationship, *i.e.*, $T = (V_T, E_T)$, where $V_T = \{v \in V \mid \pi[v] \in V\} \cup \{s\}$, and $E_T = \{(\pi[v], v) \in E \mid v \in V_T - \{s\}\}$.

Theorem 4.1 *A route $T = (V_T, E_T)$ derived at any time instance throughout MRMA is a multicast tree rooted at s .*

Proof. We prove this theorem by showing that T satisfies the properties of a multicast tree: T is acyclic, and every node $v \in V_T - \{s\}$ has exact one predecessor. It stands to reason that every node in T , except for s , has exact one predecessor because v is in T if and only if v has a non-*NIL* predecessor $\pi[v]$. We now show that T is acyclic.

It is proved by a contradiction: Let T have a cycle $c = \langle v_1, v_2, \dots, v_k \rangle$, where

$v_k = v_1$. That is, $\pi[v_i] = v_{i-1}, \forall 2 \leq i \leq k$. Without loss of generality, let us assume that the cycle occurs immediately after $\pi[v_k] \leftarrow v_{k-1}$ is done. Let us examine $m[v_i], \forall 2 \leq i \leq k-1$, right before $\pi[v_k] \leftarrow v_{k-1}$. The last update to $m[v_i]$ was done by the assignment $m[v_i] \leftarrow \min\{m[v_{i-1}], \beta(v_{i-1}) - \omega(v_{i-1}, v_i) - \gamma(v_{i-1}), \beta(v_i) - \gamma(v_i)\}$. It implies that $m[v_i] \leq m[v_{i-1}]$ at that moment. Note that even if $m[v_{i-1}]$ changes its value since then, it should never decrease. Thus,

$$m[v_i] \leq m[v_{i-1}], \forall i = 2, 3, \dots, k-1. \quad (4.1)$$

Because the cycle occurs immediately after v_k selects v_{k-1} as its predecessor, we should also have an inequality before then:

$$m[v_k] < \min\{m[v_{k-1}], \beta(v_{k-1}) - \omega(v_{k-1}, v_k) - \gamma(v_{k-1}), \beta(v_k) - \gamma(v_k)\} \leq m[v_{k-1}]. \quad (4.2)$$

By comparing the inequality (4.2) with the $k-2$ inequalities (4.1), we reach a conclusion:

$$m[v_k] < m[v_{k-1}] \leq m[v_{k-2}] \leq \dots \leq m[v_1].$$

It is a contradiction because $v_k = v_1$. As a result, T is acyclic and a multicast tree.

■

Lemma 4.1 *For any node $v \in V$ visited by control messages, $m[v] \leq \mu(s, v)$ always holds throughout MRMA.*

Proof. This lemma is proved by a contradiction: Suppose that b is the first node with $m[b] > \mu(s, b)$, and it occurs immediately after b receives a control message broadcasted from another node a . At that time instance, we should have $\pi[b] = a$

and

$$m[b] = \min\{m[a], \beta(a) - \omega(a, b) - \gamma(a), \beta(b) - \gamma(b)\}. \quad (4.3)$$

The residual energy over a maximum-residual path p_b^* from s to b is no less than that over any other path from s to b . To be more specific, it is no less than the residual energy over one particular path formed by concatenating a maximum-residual path p_a^* from s to a and the edge (a, b) . Therefore,

$$\mu(s, b) \geq \min\{\mu(s, a), \beta(a) - \omega(a, b) - \gamma(a), \beta(b) - \gamma(b)\}. \quad (4.4)$$

Because $m[b] > \mu(s, b)$, Equations (4.3) and (4.4) imply that $m[a] > \mu(s, a)$, which contradicts the choice of b as the first node for which $m[b] > \mu(s, b)$. ■

Lemma 4.2 *Given any node v reachable from s , there exists a time instance t during the execution of MRMA such that $m[v] \geq \mu(s, v)$ since time t .*

Proof. This lemma can be proved by an induction on the number of hops: Consider a node v , and let $p = \langle s = v_0, v_1, \dots, v_k = v \rangle$ be a maximum-residual path from s to v in G . We shall show that there exists a time instance such that $m[v_i] \geq \mu(s, v_i)$ thereafter, for all $0 \leq i \leq k$.

Inductive Base: After the initialization, we have $m[s] = \beta(s)$. The only path from s to s is the node s itself. By definition, $\mu(s, s) = \hat{\beta}(s) = \beta(s)$. Because $m[s]$ never decreases, $m[s] \geq \mu(s, s)$ thereafter.

Inductive Step: For the induction hypothesis, suppose that there exists a time instance such that $m[v_{k-1}] \geq \mu(s, v_{k-1})$ thereafter. There is a control message broadcasted from v_{k-1} to v whenever $m[v_{k-1}]$ is reassigned. Let t be the time instance

immediately after v receives the control message broadcasted from v_{k-1} because of $m[v_{k-1}] \leftarrow \mu(s, v_{k-1})$. At time t , we have

$$m[v] = \min\{m[v_{k-1}], \beta(v_{k-1}) - \omega(v_{k-1}, v) - \gamma(v_{k-1}), \beta(v) - \gamma(v)\}. \quad (4.5)$$

By the induction hypothesis, $m[v_{k-1}] \geq \mu(s, v_{k-1})$. Let $p' = \langle v_0, v_1, \dots, v_{k-1} \rangle$. It is clear that $\mu(s, v_{k-1}) \geq \hat{\beta}(p')$ because $\mu(s, v_{k-1})$ is no less than the residual energy over any path from s to v_{k-1} .

By replacing $m[v_{k-1}]$ with $\hat{\beta}(p')$ in Equation (4.5), the following inequality is derived:

$$\begin{aligned} m[v] &\geq \min\{\hat{\beta}(p'), \beta(v_{k-1}) - \omega(v_{k-1}, v) - \gamma(v_{k-1}), \beta(v) - \gamma(v)\} \\ &= \hat{\beta}(p). \end{aligned}$$

Because p is a maximum-residual path from s to v , $\hat{\beta}(p) = \mu(s, v)$. Therefore, $m[v] \geq \mu(s, v)$ since t . ■

Theorem 4.2 *MRMA will terminate within some finite time. Let T be a multicast tree derived by MRMA when it terminates, and T_R be T by including only those nodes in R and their ancestors. T_R is a maximum-residual multicast tree.*

Proof. In MRMA, a node v broadcasts control messages if and only if $m[v]$ is reassigned. The value of $m[v]$ never changes once $m[v] = \mu(s, v)$ (based on Lemmas 4.1 and 4.2). It implies that MRMA terminates within some finite time.

We shall prove that the three properties of a maximum-residual multicast tree hold for T_R : (1) T_R is a multicast tree rooted at s ; (2) T_R includes all of the nodes in R ; (3) $\hat{\beta}(T_R) \geq \hat{\beta}(T')$, for any multicast tree T' from s to R . The first

property follows directly from Theorem 4.1. The second property also holds, because all of the nodes in R are in T (implied by Lemma 4.2) and are not excluded by T_R . It remains to show that $\hat{\beta}(T_R) \geq \hat{\beta}(T')$.

The correctness of the third property is trivial when $R = \{\phi\}$, *i.e.*, $T = s$. To prove it for the other cases, we shall show that there exists a node $d \in R$ such that $\hat{\beta}(T_R) \geq \mu(s, d) \geq \hat{\beta}(T')$. Without loss of generality, suppose that x is a node with the minimum residual energy in T_R , *i.e.*, $\hat{\beta}(T_R) = \hat{\beta}_{T_R}(x)$. If x is a leaf, then x itself is a node in R , and we choose x as d . Otherwise, the outgoing edge of x with weight $\omega(x, y) = \max\{\omega(x, v) \mid \forall (x, v) \in T_R\}$ must be in a path from s to some leaf nodes in R . We choose anyone of them as d . Let us say that the path from s to the chosen node d in T_R is p . Then, we have $\hat{\beta}_{T_R}(x) = \hat{\beta}_p(x)$. Because x is in p , the residual energy of x over p is certainly no less than the residual energy over p , *i.e.*, $\hat{\beta}_p(x) \geq \hat{\beta}(p)$. Based on Lemmas 4.1 and 4.2, we know that $\hat{\beta}(p) = m[d] = \mu(s, d)$. We can conclude that $\hat{\beta}(T_R) \geq \mu(s, d)$.

On the other hand, we know that T' must contain a path p' from s to the chosen node d (because it includes all of the nodes in R). It implies that $\hat{\beta}(T') \leq \hat{\beta}(p')$. Note that the residual energy over p' is no larger than that over a maximum-residual path from s to d , *i.e.*, $\hat{\beta}(p') \leq \mu(s, d)$. In other words, $\hat{\beta}(T') \leq \mu(s, d)$.

■

Theorem 4.3 *A maximum-residual multicast tree T_R is an optimal solution that maximizes the minimum residual energy of all of the nodes in G .*

Proof. Let us consider a node x with the minimum residual energy in G (after the session of data packets is routed over T_R). There are two cases: (1) If $x \notin T_R$, the residual energy of x remains unchanged, and $\hat{\beta}_{T_R}(x) = \beta(x)$. T_R is an optimal

solution because the minimum residual energy of nodes in G must be no larger than $\beta(x)$ by routing on any multicast tree. (2) If $x \in T_R$, then there exists a node in T' (also in G) whose residual energy is no larger than $\hat{\beta}_{T_R}(x)$. Note that we have proved that $\hat{\beta}(T_R) \geq \hat{\beta}(T')$ for any multicast tree T' from s to R . Therefore, the minimum residual energy of nodes in G can be maximized by routing over T_R . ■

The proofs show that the route derived by MRMA is theoretically optimal. In reality, the optimality of derived routes may not hold due to the collisions of control messages or the movements of nodes. When mobility and collisions are taken into consideration, the following theorem shows that every derived route remains loop-free and, if no link breakage occurs, it converges toward an optimal solution, in the sense that $\hat{\beta}(T_j) \geq \hat{\beta}(T_i)$ if $t_j \geq t_i$, where T_j and T_i are the routes respectively derived at t_j and t_i after all nodes are visited by control messages.

Theorem 4.4 *Considering mobility and collisions, every derived route during MRMA remains loop-free, and it converges toward a maximum-residual multicast tree if no link breakage occurs.*

Proof. In MRMA, the estimate of node v changes and its predecessor is reassigned to node u if and only if $m[v] < \min\{m[u], \beta(u) - \omega(u, v) - \gamma(u), \beta(v) - \gamma(v)\}$. It implies that the estimate of a node never becomes smaller than that of any of its descendants, and that the predecessor relationship of nodes in a derived route forms directed (simple) paths. Thus, loop freedom always holds for any derived route.

Let T_j and T_i be the routes derived at t_j and t_i , respectively, after all nodes are visited by control messages. For any route T , we already proved in Theorem 4.3 that the residual energy over T is equal to the estimate of a node with the minimum residual energy, *i.e.*, $\hat{\beta}(T) = \min\{m[u] \mid \forall u \in T\}$. Because the estimate of any node

never decreases, if no link breakage occurs, the minimum residual energy over T_j is no less than that over T_i . ■

4.3 A Maximum-Residual Multicast Protocol

In this section, we develop a power-aware multicast protocol which is adaptive to dynamic network topologies and resources for large-scale mobile ad hoc networks. The Maximum-Residual Multicast Protocol (MRMP), a realization of routing algorithm MRMA, is a *pure* source-initiated on-demand routing protocol which establishes routes if and only if they are desired by sources. MRMP uses a broadcast route discovery mechanism, as used in other on-demand routing protocols [38, 47, 64, 66], with special designs to adapt to large-scale mobile ad hoc networks with energy-energy considerations. In MRMP, a route is established by autonomous decisions of intermediate nodes, instead of being determined by the source with global information. Furthermore, no node regularly maintains routes to others. No periodic control message exists for neighborhood and group management, and no control message is initialized for route repairs. A route (once established) is considered *valid* within some time interval (relative to the moving velocity) because it is determined on demand based on the network status at the time being, rather than on any information collected a priori.

Since MRMA is realized by MRMP, control messages and table entries of MRMP directly correspond to their counterparts of MRMA. There are three major stages in MRMP: Whenever a source needs a route, the source requests for a route discovery within the network so that each node decides its predecessor, as shown in Figure 4.2(a). The destination nodes and their ancestors then inform their predeces-

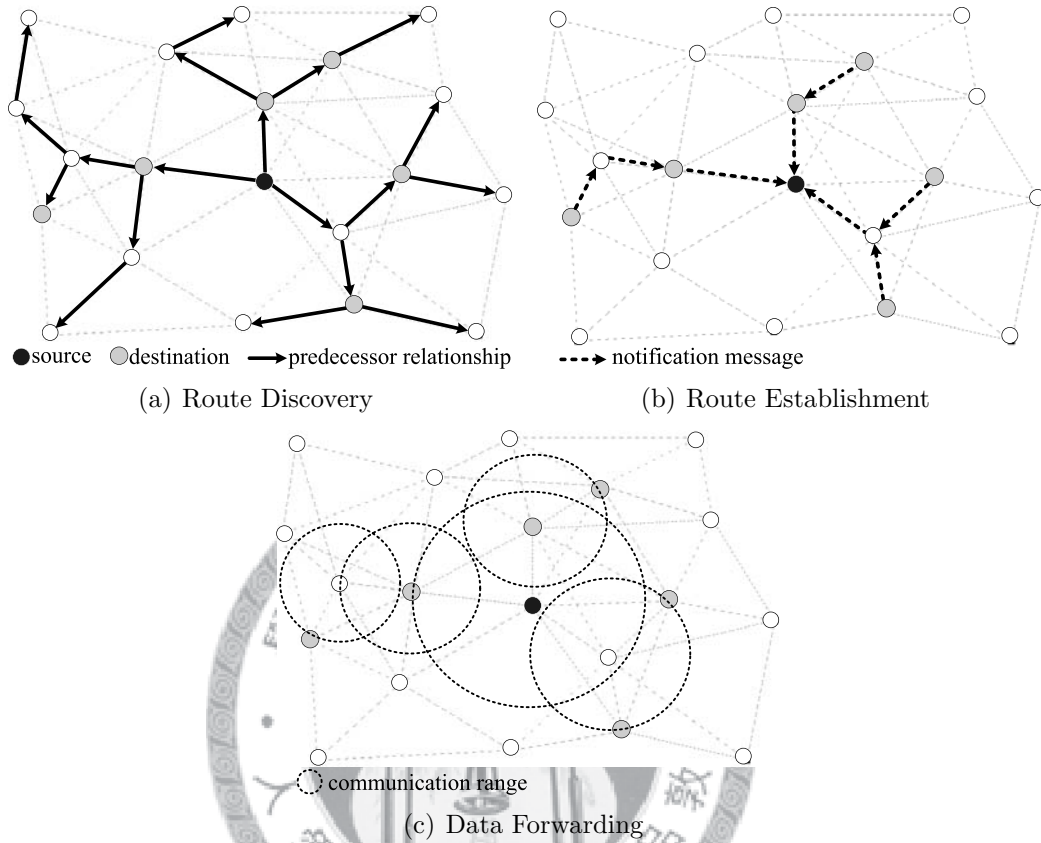


Figure 4.2: The Maximum-Residual Multicast Protocol (MRMP)

sors of the proper power levels during route establishment, as illustrated by Figure 4.2(b). Figure 4.2(c) shows the forwarding of data packets by nodes at proper power levels on the established route. In the following sections, we shall go into detailed design issues.

4.3.1 Routing Tables

In MRMP, two tables are maintained at each node: The *Group Table* and the *Route Table*. The *Group Table* is used to maintain the group information of the node, where each source represents a group. A node becomes a destination of a group if the corresponding source is added into its *Group Table*. Note that which groups a

node joins is maintained by each node itself autonomously. No control message is needed to inform others of changes in its Group Table.

The Route Table is used to record the proper power level to transmit the data packets of each session. This table consists of routing entries, where each entry is associated with a session (*i.e.*, the counterpart of an entry in MRMA). An entry remains *valid* until it is removed because of the table space limitation. The fields of an entry are as follows:

- Source ID (*i.e.*, s)
- Session Number (*i.e.*, S)
- Transmission Power
- Remaining Energy (*i.e.*, β)
- Estimate (*i.e.*, m)
- Predecessor (*i.e.*, π)
- Adjust Ratio
- Membership Flag
- Entry Status



An entry is indexed by a unique pair of **Source ID** and **Session Number**, where an example implementation of **Source ID** is the IP address of the source that originates this session, and **Session Number** can be a sequence number maintained in the source to distinguish different sessions. **Transmission Power** records the proper power level of the node to transmit the data packets of the associated session. Note that **Transmission Power** is a replacement of the role of **Next Hop** in a traditional route table, and it is set during the route establishment, as described later in Section 4.3.3. **Remaining Energy** is set as the amount of the remaining energy of the node when this entry is created (and it remains the same thereafter). **Estimate**,

Predecessor, and **Adjust Ratio**, are used to keep track of a temporary decision in route discovery and are subject to changes. **Estimate** and **Predecessor** are set according to MRMA. **Adjust Ratio** indicates the ratio of the maximum transmission power of its current predecessor such that itself can receive data packets from the predecessor successfully. The calculation of this ratio relies on some information provided by the *Physical* (PHY) layer, and it will be discussed in Section 4.3.2. **Membership Flag** indicates the relationship between the node and the session associated with this entry. This flag can be set as either **IN_GROUP** (*i.e.*, a destination of the session), **ON_TREE** (*i.e.*, an intermediate node that helps to forward the session), or **NO_RELATION** (*i.e.*, a node unrelated to the session). **Entry Status** denotes the current stage of the entry: **RTF_DISCOVERY** (*i.e.*, route discovery), **RTF_ESTABLISHMENT** (*i.e.*, route establishment), or **RTF_READY** (*i.e.*, data forwarding). Furthermore, each entry is associated with two timers, referred to as **dtimer** and **etimer**. They are used to trigger stage changes of the entry.

4.3.2 Route Discovery

A route-discovery procedure is invoked when a source has data packets to send. The source creates an entry in its Route Table with **Membership Flag** and **Entry Status** initialized as **ON_TREE** and **RTF_DISCOVERY**, respectively. The **dtimer** associated with this entry is then activated. Route discovery begins with a broadcast of a request message (REQ) from the source to all of its neighbors with its maximum transmission power (this corresponds to Line 5 of MRMA). The data frame of an REQ contains

the following fields:

```

    < source_ID, session_number, packet_size,
      number_of_packets, remaining_energy, estimate >

```

The pair of the first two fields is used to identify the REQs employed for a specific session. `packet_size` and `number_of_packets` respectively denote the number of bits in a data packet and the number of packets in the session. The last two fields are used to carry the values of the remaining energy and the estimate recorded in the associated entry.

When a node v receives an REQ from a neighboring node u , a corresponding entry is created in the Route Table of v if it does not exist (refer to Lines 7-10 of MRMA). Node v starts participating in the route discovery by setting Membership Flag as `NO_RELATION` and Entry Status as `RTF_DISCOVERY` and activating `dtimer`. The routing information is then extracted from the REQ to test whether Estimate can be increased (see Line 11 of MRMA). If it can be increased, Estimate, Predecessor, and Adjust Ratio are updated (refer to Lines 12-13 of MRMA), and then the REQ is rebroadcasted to neighbors with the last two field replaced by the Remaining Energy and the new Estimate of node v (see Line 14 of MRMA). This process is repeated until the `dtimer` expires.

One technical issue is on the determination of Estimate, which relies on the energy consumed by u to receive the session, *i.e.*, $\gamma(u)$, and to transmit the session to v , *i.e.*, $\omega(u, v)$, and the energy consumed by v to receive the session, *i.e.*, $\gamma(v)$. In order to derive the required energy amounts, a solution is to place additional information (to inform v) in the PHY header of each REQ of u . When an REQ is passed down to the *Medium Access Control* (MAC) layer, node u predicts the time

that its transceiver will spend in transmit and receive modes according to its data rate and the session size [26, 31], denoted by `txtime` and `rxtime`, respectively. When the REQ is further passed down to the PHY layer, additional information is placed in the PHY header regarding how much energy will be consumed by the transceiver in transmission for `txtime` time if its maximum power level is adopted, and that in reception for `rxtime` time, denoted by `max_tx_consumption` and `rx_consumption` respectively.

When node v receives the REQ from u , $\gamma(u)$ is namely `rx_consumption`. In a similar way, $\gamma(v)$ can be derived at node v based on the information carried in the REQ and its own device characteristics. The derivation of $\omega(u, v)$ utilizes a measurement *Received Signal Strength Indication* (RSSI) in wireless environments to measure the received signal strength [31, 40]. Because the signal strength measured at node v (denoted by P_r) is proportional to the transmission power adopted by node u (denoted by P_t) [6], `Adjust Ratio` is thus set as $\frac{P'_t}{P_t} = \frac{P_{min}}{P_r}$, where P'_t and P_{min} denote respectively the proper power level that node u should adopt and the *receive sensitivity* of node v (*i.e.*, the faintest strength of signals receivable by v). Moreover, the energy consumption is related to the adopted power level and can be calculated according to an equation, such as $\omega(u, v) = (\text{max_tx_consumption}) \times (\text{Adjust Ratio})$.

4.3.3 Route Establishment

As the `dtimer` of an entry expires, the `etimer` is activated and the `Entry Status` is set to `RTF_ESTABLISHMENT`. For those REQs (devoted to this session) that arrive late are simply discarded. Route establishment is to let each node (if needed) inform its predecessor of the `Adjust Ratio` kept in its Route Table so that the predecessor will

use a proper power level to transmit the session. Each destination or intermediate node sends exactly one reply message (RPY) to inform its predecessor. After a destination (/intermediate) node sends its RPY, it changes its `Membership Flag` to `IN_GROUP (/ON_TREE)`. The data frame of an RPY contains the following fields:

⟨ `source ID`, `session number`, `adjust ratio` ⟩

No hand-shaking mechanism is provided for multicasting or broadcasting in the MAC layer. In order to prevent the only one RPY sent by a node from collision, the destination field in the IP header of the RPY is set as the address of the predecessor, rather than the broadcast address used in REQs. In this way, each RPY is considered as a one-hop unicast packet by the MAC layer, and a hand-shaking mechanism, such as RTS/CTS [45], can be used to reserve the medium for the duration required to send the RPY, thus mitigating collision. Even if collision occurs, the MAC layer will automatically retransmit the RPY to further improve the probability of successful reception. During route establishment, a node may receive multiple RPYs, and the maximum value of `Adjust Ratio` must be kept. When the `etimer` expires, the field `Transmission Power` in the corresponding entry is set as its maximum transmission power level multiplied by the kept `Adjust Ratio`.

One technical issue is on the existence of asymmetric links, *e.g.*, the edge between x and y in Figure 4.1, so that the RPY of a node can not be sent back to its predecessor. The implementation of MRMA should be revised by considering the support of some underlying layers, such as the *Logical Link Control* (LLC) layer, which maintains the *Address Resolution Protocol* (ARP) table for mapping IP addresses to MAC addresses [45]. The ARP table is taken as an example because the table caches only the MAC addresses of those nodes reachable by one hop.

Consider the following scenario as an example implementation: Let a node v receive an REQ from a neighboring node u . The REQ is dropped as usual if u is not a better choice (refer to MRMA); otherwise, v tries to resolve the address of u by looking up its ARP table and broadcasting an ARP packet to its neighbors if necessary. If the addresses is resolved, then v updates its predecessor to u ; otherwise, the routing entry remains unchanged because u may not be directly reachable by v .

4.3.4 Data Forwarding

When the timer of an entry expires, the Entry Status is set as RTF_READY and the routing entry is considered *active*. When an entry of a source is active, the source begins to transmit the data packets of the associated session using the Transmission Power in the entry.

When a node receives a data packet of a session, the node checks up its Route Table to determine the next action: (1) If no entry associated with this session exists, the data packet is simply dropped. (2) If the associated entry is not active yet, the data packet is queued, and its further action is delayed until the entry becomes active. (3) If the associated entry is active already, the data packet is transmitted with the corresponding Transmission Power. Note that forwarding is not needed by this node if the Transmission Power is set as 0 (default value as the entry created). (4) Furthermore, if the Membership Flag of the associated entry is set as IN_GROUP, then the data packet should be passed to the upper layer and considered as being received by a destination.

One technical issue is on the receiving of multiple duplicates of a data packet by a node. Such a phenomenon is referred to as *overhearing* in wireless communica-

tion. *Overhearing* may result in extra energy consumption and has yielded a serial of research. A typical approach is to turn the transceiver into sleep mode during the required by other nodes to complete their communication [90]. The duration is usually indicated in the *Network Allocation Vector* (NAV) of RTS/CTS frames [31]. Nevertheless, RTS/CTS (in 802.11 MAC) is devoted to unicast only, neither to multicast nor to broadcast [31]. To mitigate overhearing in multicast, a similar concept is adopted in the network layer. A *Announce to Send* (ATS) control message (with the duration indicated in its frame) is broadcasted immediately before the to-be-sent data packets so that nodes can decide to receive the incoming data packets or go into sleep mode. Note that the approach relies on the support of sleep mode, and thus ATS is considered as an optional control message.

4.4 Performance Evaluation

The purpose of this section is to evaluate the applicability of MRMP in terms of scalability and mobility with essential performance metrics adopted generally for routing protocol evaluation.

4.4.1 Experimental Setups and Performance Metrics

We implemented MRMP over NS2 (version 2.31) [26], a network simulator popularly used in the evaluation of routing and multicast protocols, and conducted extensive simulations. The main objective of the simulations is to demonstrate the performance of MRMP in large-scale mobile ad hoc networks. NS2 provides only omnidirectional antennae in its current version, and the real-world RF radiation may

not behave so ideally with disk-shaped radiation patterns. We implemented a user-configurable antenna module to evaluate the capability of MRMP with asymmetric communication links (resulted from irregular radiation patterns). The experimental results of MRMP with omnidirectional and irregular antennae are respectively denoted by MRMP_0 and MRMP_r . We also conducted simulations on MAODV [66], a multicast protocol that was intended for use in mobile ad hoc networks by the IETF MANET Working Group [67]. The adopted MAODV implementation was implemented by Thomas Kunz et al. [92]. Because MAODV only supports the use of symmetric links [66], we did not evaluate it with irregular antennae. However, for fairer comparison purposes, we implemented a power adaptation module in MAODV so as to use proper power for transmission. The experimental results of MAODV with and without power adaptation are respectively denoted by MAODV_F and MAODV_P . We refer interested readers to some performance studies of several representative multicast protocols in [44, 48, 83].

The simulations were studied over networks with different numbers of nodes. Nodes were randomly placed in a $500 \times 500 \text{ m}^2$ rectangular area and then kept moving freely within this area. The Random Walk Mobility Model [51] with various maximum velocities was adopted for the mobility pattern, where the scenario generator of NS2 was used to generate topology scenarios. Each node in the study was equipped with an 802.11 adaptor in the ad hoc mode, and its specifications is listed in Table 4.1, referred to Intel[®] PRO/Wireless 2011 LAN PC Card [2]. Unlisted specifications had the default values of NS2. Each node had a battery with a capacity of 2160 joules (being able to be idle for 5 hours).

In this study, we measured the radiation pattern of an F-shaped antenna (embedded in the Chipcon CC2420 RF transceiver [3]) by Ansoft HFSS [1], which is

Specification	Setting
Network Standard	IEEE 802.11b
Antenna Type	Omnidirectional or Irregular
Frequency	2.4GHz ISM Band
Data Rate	11Mbps
Media Access Control	CSMA/CA
Propagation Model	TwoRayGround
Max Transmission Power	18dBm
Receiver Sensitivity	-62dBm
Average Communication Range (at Max Power)	100m
Average Carrier Sense Range (at Max Power)	120m
Transmitter Power Consumption	6.0 watt (= $0.5A \times 12V$)
Receiver Power Consumption	2.04 watt (= $0.17A \times 12V$)
Idle Power Consumption	0.12 watt (= $0.01A \times 12V$)

Table 4.1: A wireless adaptor specification

one of the most popular and powerful software used for antenna design/analysis. An irregular radiation pattern is often represented as *gain values* of signals with angles relative to directions (along that the signal radiates). Figure 4.3 shows the footprint details of the antenna and the gain values of its radiation pattern. The sampled gain values were stored in a table for quick look-up, and *linear interpolation* was used to estimate those between the finite sampled values. NS2 assumes that nodes are placed on the same *xy*-plane. The average and the standard deviation of the gain values (after normalization) on the selected plane were 0.839 and 0.1, respectively. The average communication range (at the maximum transmission power) was set as 100m (for comparison with the omnidirectional case), and the maximum and minimum communication distances (according to the gain values) were about 141m and 76m, respectively.

For each topology scenario, the first 100 nodes were chosen as sources. All of the nodes joined the only *one* multicast group at the beginning of the simulation and remained as members throughout the whole simulation. This setup is because it was simply impossible to run NS2 simulations of MAODV over networks of a certain scale within a reasonable amount of time. Control messages for membership

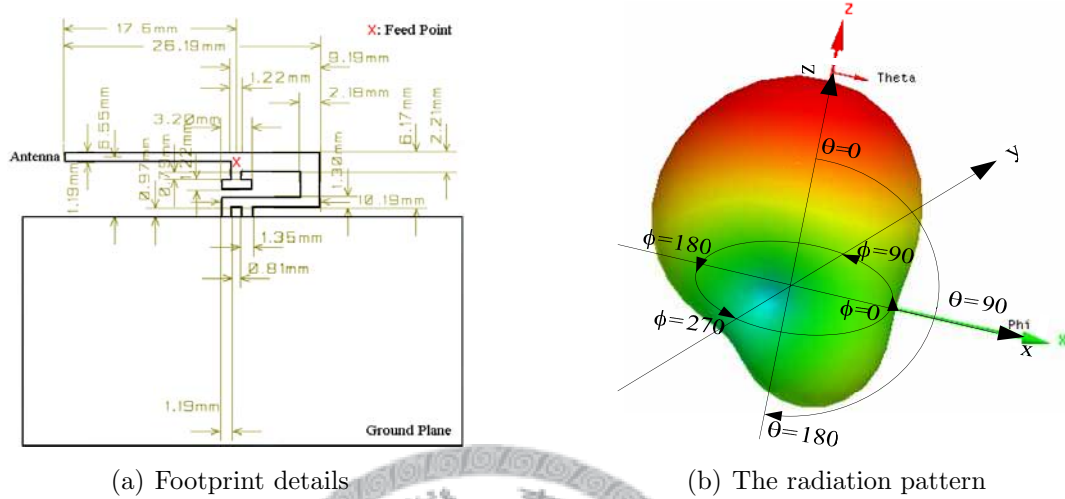


Figure 4.3: An F-shaped antenna

maintenance in MAODV would lead to traffic flooding in the simulations when the number of nodes or groups in the network was over some value. Because of the single group setting, all of the remaining nodes were assigned as destinations of each multicast. Each source originated a 1MB media stream with a data rate of 0.5 Mbps. Each media stream was composed of 500 UDP packets that were 2048 bytes in length. Each source was randomly selected in the 100 sources to multicast a media stream per minute until there was one node that exhausted its energy. The *uniform distribution* was adopted for the random source selection. Each topology and traffic scenarios were generated for a 5-hour simulation. In order to evaluate the protocol scalability with respect to the network size, simulations were conducted over static networks of 100 to 1000 nodes. The maximum node velocity was varied from 1m/s to 10m/s in networks of 100 nodes¹ to evaluate the protocol applicability to the node mobility. `dtimer` and `etimer` of MRMP were both set as $(0.1 \times |V|)$ ms so that there was sufficient time for a node to exchange $(0.4 \times |V|)$ REQs with its neighbors or for $(0.15 \times |V|)$ mutual-interfere nodes to send their RPYs sequentially. ATS was disabled in the simulations because the current 802.11 module of NS2 does

¹Running large-scale simulations in NS2 was very time-consuming. Simulating a network of 200 nodes (or more) that kept moving for 5 hours cannot terminate in some reasonable time.

not support sleep mode. The parameter setting of MAODV was the same as that in [92].

Four performance metrics were considered in routing protocol evaluation. Beside the consideration of energy efficiency, three metrics were also adopted based on popular routing metrics, as suggested by the IETF MANET Working Group [22]:

- **Network Lifetime:** measured by the average number of data packets (excluding duplicates) a node can transmit and receive before the first node exhausts its energy. This demonstrates the effectiveness of a routing protocol (and its routing metric) in the prolongation of network lifetime.
- **Delivery Ratio:** the ratio of the number of data packets successfully delivered to the destinations to the number of data packets supposed to be received. It shows the capability of a routing protocol to successfully deliver data packets to destinations.
- **Control Overhead:** the average number of control bits transmitted per data bit delivered. The control overhead includes not only the control messages in the network layer but also those in the MAC layer. This measures the bit efficiency of a routing protocol in expending control overhead to delivery data.
- **Propagation Delay:** the average time delay required for a data packet to be propagated from the source to a destination, including the route discovery period. It presents the efficiency of a protocol in data routing.

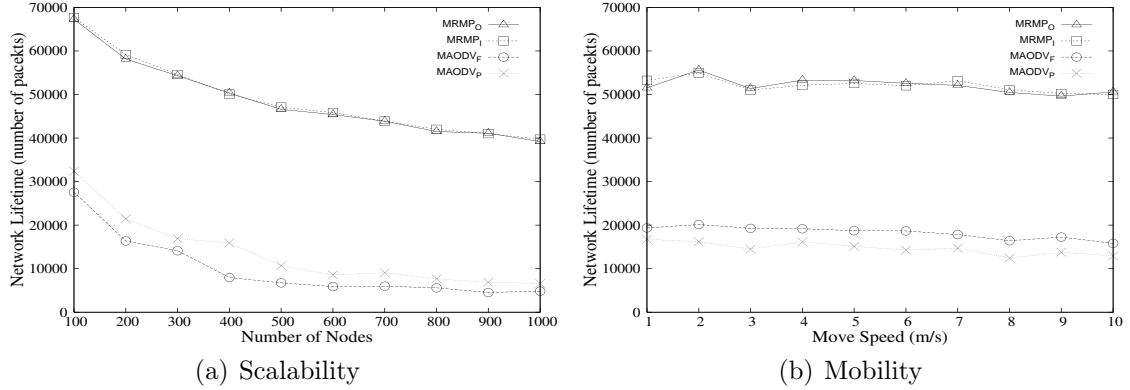


Figure 4.4: Network Lifetime

4.4.2 Experimental Results

Figure 4.4 shows the network lifetime with respect to the network size and node mobility. As shown in Figure 4.4(a), the network lifetime decreases as the number of nodes increases, which might violate the intuition that dense networks might have more alternative routing paths such that more energy-efficient routes can be found by MRMP to prolong the network lifetime. Such a phenomenon is due to the overhearing problem. As mentioned early, the current 802.11 module of NS2 does not support the sleep mode. This result also shows the strong motivation in energy conservation research in the MAC layer. In the simulations, MAODV receives more duplicated data packets, because every node uses much larger transmission power and more nodes lie in its communication range as the node density increases. MAODV with power adaptation indeed saves some unnecessary energy, but its routing metric still encourages it to use large power for transmission. The results show that MRMP₀ outperforms MAODV_F and MAODV_P in terms of network lifetime by 2 to 9 times, as the network size increases from 100 to 1000 nodes, where the overhearing problem exists. The performance difference between MRMP₀ and MRMP₁ is within a small margin. We observed that the shapes of radiation patterns have no significant impact on the performance of MRMP, when most of the RPYs of

nodes can be delivered to their predecessors successfully. However, some extreme radiation patterns (*e.g.*, those of highly-directional antennae) would result in unacceptable performance and might be unsuitable for multicasting. Note that MRMP is very effective in the prolonging of the network lifetime, even with the overhearing problem. It is because MRMP attempts to route data packets so that the energy consumption is balanced among the nodes in proportion to their remaining energy.

Furthermore, as we can see in Figure 4.4(b), the velocity of a node has no significant impacts on the network lifetime. The main reason behind the observation is that mobility does not affect the number of packets which the given battery capacity can afford for a node to receive. However, MAODV may react to link breakages by initiating control messages (and thus consuming extra energy) for route repairs, so the network lifetime slightly decreases as the velocity increases. Route repairs occur more frequently in MAODV_P because of its smaller communication ranges, compared with MAODV_F. In addition, we have to point out that the network lifetime dramatically decreases as the velocity increased from 0m/s to 1m/s, compared with the results in Figure 4.4(a). This is mainly resulted from the fact that nodes may leave the communication ranges during data transmissions (because communication ranges is fixed once the multicast tree was constructed), and thus fewer nodes apportion the energy consumed for the transmissions. Besides, nodes might receive more duplicated packets and thus consume more energy once they run into the communication ranges of other nodes. Figure 4.4(b) shows that MRMP₀ is over 2.6 times more effective than MAODV_F and MAODV_P in the prolongation of the network lifetime when the maximum velocity is over 1m/s.

Figure 4.5 shows the impacts of the network size and node mobility on the delivery ratio. As shown in Figure 4.5(a), MRMP scales very well with respect to

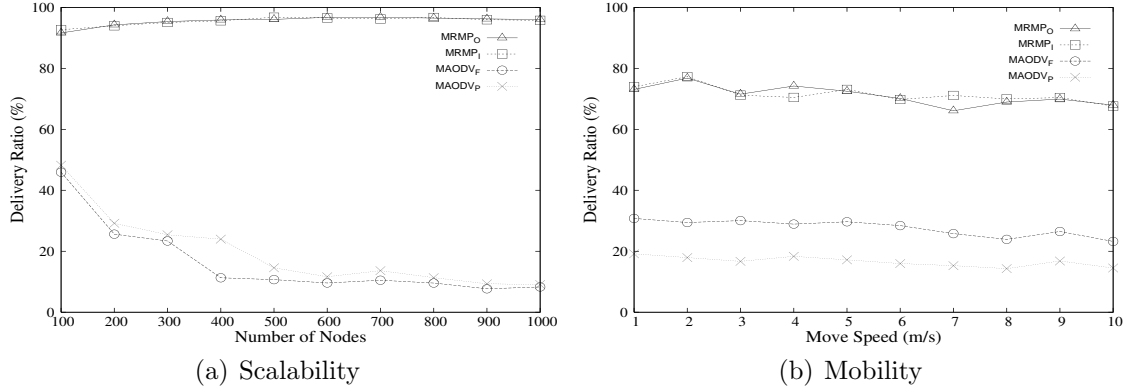


Figure 4.5: Delivery Ratio

the data delivery. The delivery ratio is roughly 96%. Unlike MRMP, MAODV is very sensitive to the network size. The delivery ratio drops abruptly around from 47% to 8% as the number of nodes increases from 100 to 1000 because of a high collision probability. Note that MAODV_F always and MAODV_P probably use the maximum power levels for data transmission, which causes heavy potential collisions. Another reason is that packets are dropped once buffer overflow occurs. It is usually harder for a node to occupy the medium when more nodes are competing for the medium (within the communication ranges of each other). We also observed that periodical messages used in MAODV for neighbor and group maintenance result in serious collisions. This also supports the disuse of periodical control messages in routing protocols for large-scale networks. In Figure 4.5(b), the delivery ratios slightly decrease as the velocity increases. It is because nodes are more likely to outrun communication ranges. This is also the reason why MAODV_P is inferior to MAODV_F in terms of data delivery. We have to point out that the delivery ratio of MRMP decreases significantly to 71%, compared with the counterpart in Figure 4.5(a). It is because MRMP uses the exact power for data transmission, and it could result in the outrunning problem. We observed that the delivery ratio can be much increased by increasing the transmission power of each node slightly. However, how much power to increase is, in fact, related to the moving pattern of the application

scenario, and further study on moving patterns could be a good research topic.

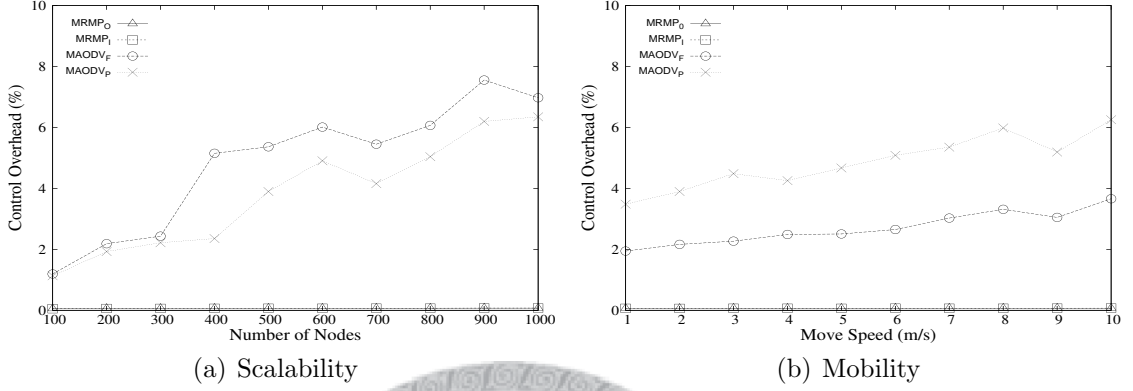


Figure 4.6: Control Overhead

Figure 4.6 shows that MRMP is very efficient in terms of control overhead in data delivery, unrelated to the network size or the node mobility. It is because MRMP does not depend on any periodical control messages for neighbor or group maintenance. Control messages are employed only if a route is desired to send data packets. The control overhead of MRMP is less than 0.08% of the entire traffic, as shown in Figure 4.6(a) and 4.6(b). In Figure 4.6(a), the control overheads of both MAODV_F and MAODV_P are much heavier than that of MRMP₀, and they increase substantially in general, as the number of nodes increases. Nodes might produce more control messages in a dense network than in a sparse network. The main reason for such high overheads is, nevertheless, the low delivery ratios shown in Figure 4.5(a), and the control overhead can be up to 7.6%. Because of a similar reason, the control overheads of MAODV_F and MAODV_P in Figure 4.6(b) increase as the delivery ratios decrease. Moreover, MAODV_P has higher control overhead than MAODV_F because of its lower delivery ratio and more vulnerable links.

Figure 4.7 shows that MRMP₀ is over 4 times more efficient than MAODV_F and MAODV_P in data routing. This result might be contrary to the intuition that MAODV ought to be more efficient than MRMP because MAODV attempts to min-

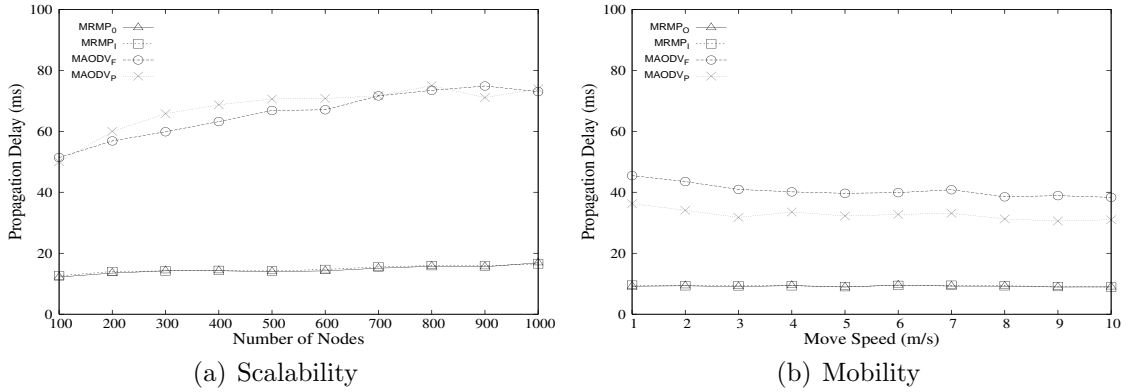


Figure 4.7: Propagation Delay

imize the number of hops. The reason for the phenomenon is that carrier sensing is done by a node to compete with its neighbors on the medium usage before the beginning of a transmission. A large communication range implies a potentially large number of competitors in the network. Furthermore, control messages are usually treated as high priority packets and inserted in the beginning of the priority queue. As a result, periodic message broadcasting for neighbor maintenance in MAODV could greatly delay the data propagation. Moreover, the propagation delay of MAODV seems not increasing with the node density so much as expected, as shown in Figure 4.7(a). It might be due to the fact that older packets are dropped because of buffer overflow and not taken into consideration in the performance measurement. Thus, propagation delay will not increase unlimitedly as node density increases. In Figure 4.7(a), the average time required by MRMP (/MAODV) to deliver a packet to a destination is about $15ms$ ($/66ms$). Figure 4.7(b) shows that the propagation delay tends to decrease as the velocity increases. This is because a node is more likely to outrun the communication range at a higher velocity, and it might result in routing tree breakages. The breakages may make nodes that are far away from the source difficult to receive the data packets. The average time in data propagation is consequently under estimated. This is also the reason why MAODV_P is superior to MAODV_F. The propagation delay of MRMP decreases not so much

because of its high delivery ratio. In Figure 4.7(b), the average propagation delays of MRMP_0 , MAODV_F , and MAODV_P are about $9ms$, $41ms$, and $33ms$, respectively.

4.5 Summary

This chapter proposes a power-aware routing protocol, namely the Maximum-Residual Multicast Protocol (MRMP), in the maximization of the minimum residual energy of nodes in large-scale mobile ad hoc networks. MRMP is a source-initiated on-demand multicast protocol, where no periodic control message is needed in the information collection of the network topology and remaining energy. Each multicast tree for a multicast is derived based on the autonomous decision of each individual intermediate node and is proved to be theoretically optimal. MRMP has demonstrated itself being effective and efficient in power-aware routing over NS2 based on parameter setting of Intel[®] wireless devices [2] and performance metrics concerned by the IETF MANET Working Group [22]. The proposed distributed methodology is also applicable to various related optimization problems (such as the minimization of the total energy consumption of any path from a source to a destination) and provides useful insights when network resources (such as bandwidth) might change over time.

Chapter 5

Bus-Layer Minimization

This chapter explores real-time task scheduling with chain-based precedence constraints in embedded systems with multi-layer buses¹. The objective is to minimize the total cost contributed by the needed bus layers without any violation of timing constraints. The contributions of this work start with fundamental but negative results on the \mathcal{NP} -hardness of the target problem and its inapproximability ratio. To be more specific, we show that, unless $\mathcal{P} = \mathcal{NP}$, it is not possible to have any approximation algorithm with an approximation ratio better than 1.5. A polynomial-time optimal algorithm, based on dynamic programming, is then proposed for a restricted case, when tasks only have unit execution times, and any communication delay is of one time unit. The algorithm is later extended as a pseudo-polynomial-time algorithm for general cases when tasks have arbitrary execution times, communication delay can be of any time units, any selected task and/or communication can be preemptive/non-preemptive, and other timing constraints/objective functions are considered. The proposed algorithm was evaluated against a list-scheduling algo-

¹Note that there do exist many applications with chain-based precedence constraints such as JPEG and MPEG decoding [3, 4].

rithm [75], over an AMBA-based system topology [53] with task generated by TGFF [80], to provide better insights to this work.

The rest of this chapter is organized as follows: Section 5.1 describes the system model under consideration and defines the problem. In Section 5.2, fundamental but negative results of this problem are first presented. Optimal algorithms with restricted and general cases are then presented in Sections 5.3 and 5.4. Simulation results and analysis are reported in Section 5.5. Section 5.6 is the conclusion.

5.1 System Model and Problem Formulation

5.1.1 System Model

A multi-layer bus system consists of heterogeneous processing elements that communicate with one another via multi-layer buses (or traditional buses), such as those based on the Advanced Micro-controller Bus Architecture (AMBA) [53]. Each bus has one or more layers, and each bus layer (similar to a traditional bus) is capable of data communication. Multiple buses of such a system are connected by one or more bridges (or switches), as shown in Figure 5.1. For the simplicity of presentation, this work focuses its discussions on multi-layer bus systems in which every processing element, except the shared memory and bridges, is a processor. However, the to-be-proposed methodology could be applied to systems with any processing elements that require based interconnection. Multiprogramming is supported on every processor, where *tasks* could be pending for execution. Each processor has multiple half-duplex ports, and each of the ports is connected with a bus layer. A processor is allowed to launch one *transaction* onto a bus layer (or receive one from

a bus layer) at a time. A shared memory unit consists of multiple interleaved banks, and each of them is connected to a bus layer to allow for simultaneous accesses. A bridge is a crossbar that supports simultaneous data exchanges among buses.

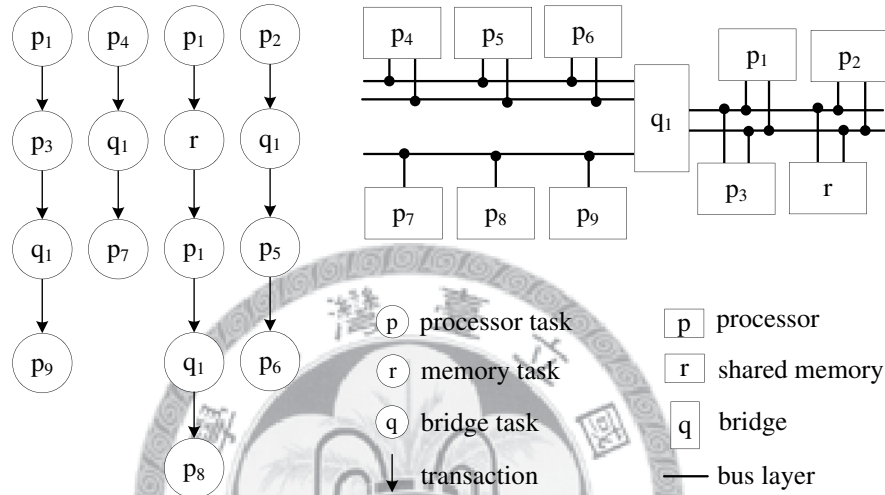


Figure 5.1: The multi-layer bus architecture

An application program consists of multiple tasks running on different processors. There is a precedence constraint between tasks, and their precedence relationship form a *precedence graph*. Nodes and edges denote tasks and their precedence relationship, respectively. Tasks running on processors are referred to as *processor tasks*. Tasks running on a shared memory unit denote memory access activities and are referred to as *memory tasks*. When a transaction is crossing a bridge, it is assumed to have a corresponding task running on the bridge, referred to as a *bridge task*. If two tasks run on two different processors (or bridges/memory), and there is an edge between them, then the edge denotes a transaction. There is a communication delay for a transaction. If the two tasks run on the same processors, then the edge between them denote a *null transaction*, *i.e.*, a transaction without any delay. Tasks are associated with deadlines (that could be ∞). In this dissertation, we are interested in precedence graphs with a constant number of chains (referred to as *precedence chains*), as shown in Figure 5.1. We first assume that all of the tasks and

transactions are preemptive in scheduling to derive the basic results, and we shall show how to extend the results for non-preemptive scheduling (Section 5.4.2).

5.1.2 Problem Definition

In this chapter, we are interested in the minimization of the needed bus layers of a given multi-layer bus system and the satisfaction of its deadline and precedence constraints. The system model under consideration can be formulated as follows:

An application program is represented by a directed graph G , in which each node u and edge (u, v) of G denote a task and a transaction between their corresponding nodes u and v , respectively. A system topology is represented by a hypergraph H , where a node x denotes a processor/memory/bridge, and a hyper-edge (x_1, x_2, \dots, x_k) denotes the bus connecting the k nodes². Every task $u \in G$ is pre-allocated to a processor/memory/bridge $x \in H$ by a function $\rho(u)$, and u is associated with two non-negative integers $e(u)$ and $d(u)$ to denote its execution time and deadline, respectively. Note that the execution time of a memory (or bridge) task represents the time required for the corresponding memory access (or bridge crossing). A transaction (u, v) is a null transaction if $\rho(u) = \rho(v)$; otherwise, it is a transaction with a non-negative delay $f(u, v)$. The corresponding bus is denoted as $\beta(u, v)$.

A *schedule* is a mapping of the tasks and transactions of an application program onto a given system topology. A schedule is *valid* if the following four conditions are true: (1) No two processor tasks run on the same processor in the same time unit, (2) the number of transactions being active on the same bus in the

²A hypergraph is a generalization of a graph, where an edge can be incident on more than two nodes [32].

same time unit is no more than its available layers, (3) no task or transaction could run unless all of its preceding tasks/transactions finish, and (4) a bridge task and its two (adjacent) transactions must run one after another without any interruption. The first two conditions are to reflect the constraints in physical resource usage of processors and buses. The third one is the enforcement of the given precedence constraints. The fourth one reflects the physical constraints of bridges. Note that there is no condition regarding memory tasks. A valid schedule is *feasible* if the maximum *tardiness* of tasks is bounded by a given threshold T_{max} . The tardiness of a task u in a schedule S is the amount of time that its completion time exceeds its deadline: That is $\max_{\forall u \in G} \{\max(0, S(u) + e(u) - d(u))\}$, where $S(u)$ denotes the starting time of task u in schedule S .

The Tardiness-Bounded Layer Minimization Problem

Instance: Consider a precedence graph G with a constant number of chains to be scheduled on a given system topology H . Each task $u \in G$ is associated with an execution time $e(u)$ and a deadline $d(u)$, and it is pre-assigned to a node of H by a function ρ . Each transaction $(u, v) \in G$ is associated with a delay $f(u, v)$ and a bus $\beta(u, v)$ (if it is applicable). Let each layer of a bus b in H require a positive cost $\omega(b)$ and T_{max} be a given tardiness threshold.

Objective: The objective is to find a feasible schedule S with $\max_{\forall u \in G} \{\max(0, S(u) + e(u) - d(u))\} \leq T_{max}$ such that $\sum_{\forall b \in H} \omega(b) \times |b|$ is minimized.

5.2 A $(1.5 - \epsilon)$ -Inapproximability Result

Before solutions to the tardiness-bounded layer minimization problem are presented, we shall show some important properties of the problem, namely its \mathcal{NP} -hardness and inapproximability ratio. We first prove the \mathcal{NP} -hardness of the problem by a reduction from the decision version of the *three-chain two-processor problem*, which is known to be \mathcal{NP} -complete [82], and then show by a gap-reduction that this problem cannot be approximated in polynomial time with a ratio better than 1.5, unless $\mathcal{P} = \mathcal{NP}$.

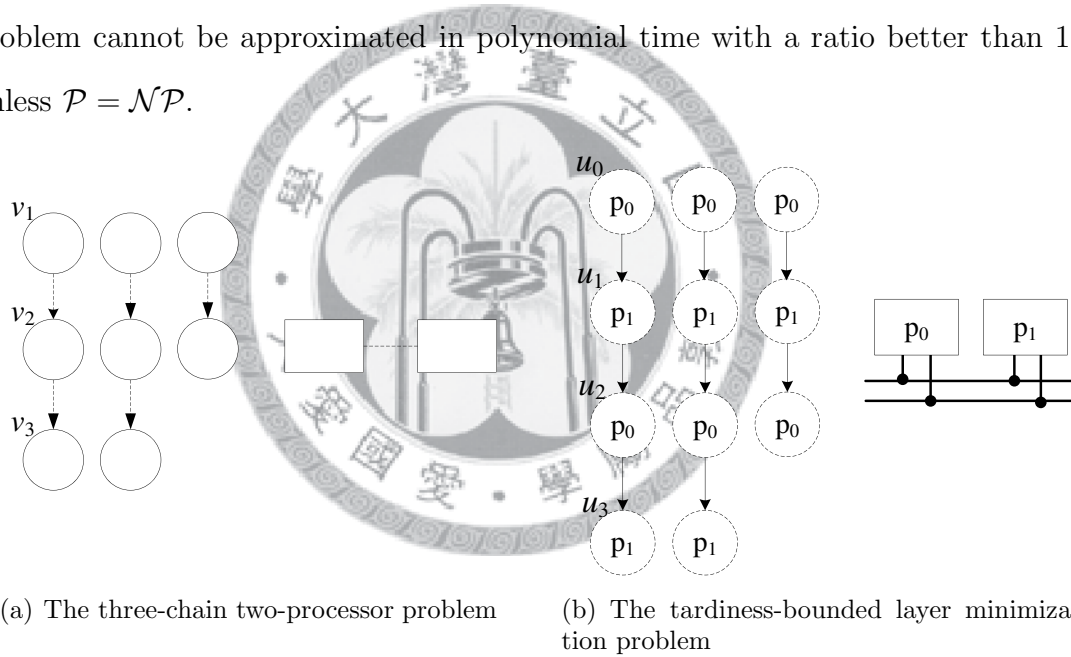


Figure 5.2: A reduction example

Lemma 5.1 *The tardiness-bounded layer minimization problem is \mathcal{NP} -hard.*

Proof. The input instance of the three-chain two-processor problem has three precedence chains G' to be scheduled on two identical processors, where any task $u \in G'$ can run on any one of the processors for $e'(u)$ time, provided the precedence order is satisfied. Note that the communication delay between the two processors is ignored in the problem. The problem output is *YES* if and only if there exists a

schedule for G' with the *makespan* (*i.e.*, the completion time) no more than a given threshold t_{max} .

Given an instance $\langle G', e', t_{max} \rangle$ of the three-chain two-processor problem, as shown in Figure 5.2, we shall show how to construct, in polynomial time, an instance $\langle G, H, \rho, \beta, e, d, f, \omega, T_{max}=0 \rangle$ of the tardiness-bounded layer minimization problem such that G' has an in-time schedule if and only if G has a zero-tardiness schedule on a system topology with the cost no more than 2ω . The construction procedure is as follows: For every chain $c' = (v_1 \rightarrow v_2 \rightarrow \dots \rightarrow v_k)$ of G' , a corresponding chain $c = (u_0 \rightarrow u_1 \rightarrow \dots \rightarrow u_k)$ of G is created. The corresponding parameters of c are set based on those of c' . The execution time and deadline of every task are set as 0 and t_{max} , respectively, *i.e.*, $e(u_i) = 0$ and $d(u_i) = t_{max}$, $\forall u_i \in c$. The communication delay of a transaction (u_{i-1}, u_i) is set as the execution time of task v_i , *i.e.*, $f(u_{i-1}, u_i) = e'(v_i)$, $\forall (u_{i-1}, u_i) \in c$. Let H be a system topology that consists of two processors p_0 and p_1 connected by a multi-layer bus b , where each layer costs ω . The tasks of c are assigned to the two processors in an alternative way, *i.e.*, $\rho(u_i) = p_{(i\%2)}$, $\forall u_i \in c$. Since two consecutive tasks of a chain are assigned to different processors, all the transactions must be done over the bus b , *i.e.*, $\beta(u_{i-1}, u_i) = b$, $\forall (u_{i-1}, u_i) \in c$.

To complete the proof, we shall show that a zero-tardiness schedule S on H with cost $\leq 2\omega$ can be used to derive an in-time schedule S' to G' in polynomial time, and vice versa. Because each task of G has zero execution time, and every transaction (u_{i-1}, u_i) corresponds to a task v_i of G' , the assignment of (u_{i-1}, u_i) on a bus layer in S implies that the corresponding task v_i is assigned to run on the corresponding processor with the same amount of time in S' . If all of the tasks in S can finish their executions no later than t_{max} on H with cost $\leq 2\omega$, then there

exists one corresponding schedule S' with the makespan no more than t_{max} . With similar argument, the other direction can be shown. Since the three-chain two-processor problem is \mathcal{NP} -complete, we conclude that the tardiness-bounded layer minimization problem is \mathcal{NP} -hard. ■

Theorem 5.1 *The tardiness-bounded layer minimization problem has no $(1.5 - \epsilon)$ -approximation algorithm, for any $\epsilon > 0$, unless $\mathcal{P} = \mathcal{NP}$.*

Proof. We prove this theorem by contradiction. Suppose that there exists a polynomial-time α -approximation algorithm \mathcal{A} for the tardiness-bounded layer minimization problem, for some ratio $\alpha < 1.5$. We shall then show how to use the hypothetical algorithm \mathcal{A} to solve the three-chain two-processor problem. We have proved in Lemma 5.1 that G' has an in-time schedule S' if and only if G has a zero-tardiness schedule S on H with cost $\leq 2\omega$. If G' has an in-time schedule, then \mathcal{A} will output a zero-tardiness schedule for G on H with cost $\leq \alpha \times 2\omega$, because \mathcal{A} is an approximation algorithm with a ratio α . Otherwise, \mathcal{A} will output a zero-tardiness schedule with more than two layers or fail to output a zero-tardiness schedule, depending on whether any solution exists or not. In that case, the cost is at least 3ω . It implies that \mathcal{A} can be used to decide whether G' has an in-time schedule if $\alpha \times 2\omega < 3\omega$, *i.e.*, $\alpha < 1.5$. Hence, unless $\mathcal{P} = \mathcal{NP}$, the tardiness-bounded layer minimization problem has no $(1.5 - \epsilon)$ -approximation algorithm, for any $\epsilon > 0$. ■

5.3 A Dynamic-Programming Approach

5.3.1 A Basic Algorithm

In this section, we first propose a dynamic-programming algorithm for a restricted case of the tardiness-bounded layer minimization problem: Let each task be of one unit execution time (called a *unit task*), and every transaction be associated with one unit of communication delay (called a *unit transaction*). We assume that there is only one multi-layer bus (but may be multiple traditional buses, considered as single-layer buses) in the system. We will later extend the result to general cases in the following sections. Before we proceed with further discussion, some terminology is defined first:

Consider a precedence graph G of multiple precedence chains to be scheduled on a given system topology H . A *partial chain* of a precedence chain is one by removing all of the preceding and succeeding edges/nodes of some specified nodes or edges. Note that a partial chain may end and/or start with an edge. A chain of no edge and node, *i.e.*, a *null chain*, is a partial chain of any precedence chain, and a precedence chain is a partial chain of itself. A *partial graph* consists of partial chains. A partial graph G_p is called a *prefix* of G if all of the preceding nodes/edges of a node u in G are also in G_p , when u is in G_p . For G_p , there is one unique *suffix* G_s of G such that $G = G_p \cup G_s$ (*i.e.*, a prefix and its suffix are complementary graphs). A task or transaction is *available* with respect to G_p if it can be executed right after all of the tasks/transactions in G_p are done. Let \hat{U} denote the maximum set of the tasks/transactions that are available with respect to G_p . An example $\hat{U} = \{(u_1, u_2), v_2\}$ of a prefix G_p is shown in Figure 5.3. A set U of tasks/transactions available with respect to G_p is referred to as *valid* with respect to

G_p if they can be scheduled at the same time without violating the four conditions (defined in Section 5.1.2). Let $\lambda(U)$ denote the number of unit transactions of U that need to be transferred via the bus b , *i.e.*, $\sum_{\forall (u,v) \in U} \{1 \mid \beta(u,v) = b \ \& \ f(u,v) = 1\}$. Let $A(G_p)$ denote the set of all the possible sets U valid with respect to G_p . In the example of Figure 5.3, $A(G_p) = \{\{v_2\}, \{v_2, (u_1, u_2)\}\}$.

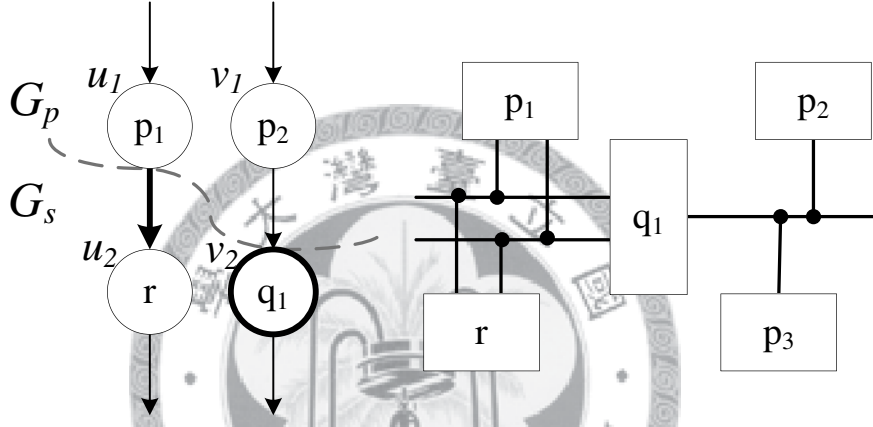


Figure 5.3: An illustration of terminology

The to-be-proposed algorithm is based on dynamic programming, as follows: Let $L(t, G_s)$ be the minimum number of layers needed by a suffix G_s of a given precedence graph G from time t , where the corresponding prefix G_p is already scheduled.

$$L(t, G_s) = \begin{cases} 0, & \text{if } G_s = \phi; \\ \infty, & \text{else if } A(G_p) = \phi \text{ or } \max_{\forall u \in \tilde{U}} \{t + e(u) - d(u)\} > T_{max}; \\ \min_{\forall U \in A(G_p)} \max(L(t+1, G_s - U), \lambda(U)), & \text{otherwise.} \end{cases}$$

The objective is to derive $L(0, G)$, *i.e.*, the minimum number of layers needed to schedule G from $t = 0$. $L(t, \phi)$ is equal to 0 for any $t \geq 0$ because the scheduling of the null graph does not need any layer. Given a suffix G_s , let G_p denote its complementary prefix, and U be any set in $A(G_p)$. When $A(G_p) = \phi$, there is no

set valid with respect to G_p and thus exists no valid schedule. The number of needed layers is set as ∞ . Moreover, if there exists a task $u \in \hat{U}$ that fails to meet the tardiness bound T_{max} at time t , i.e., $t + e(u) - d(u) > T_{max}$, then the number of needed layers is also set as ∞ . Otherwise, the number of layers needed to schedule U at t depends on the number of transactions that need to be transferred via bus b , i.e., $\lambda(U)$. If U is scheduled at t , then the rest of G_s , i.e., $(G_s - U)$, has to be scheduled from $t + 1$. As a result, the number of layers needed to schedule G_s from t (with the scheduling of U at t) is the maximum of $L(t + 1, G_s - U)$ and $\lambda(U)$. By considering all of the possible sets valid with respect to G_p , the best is selected as $L(t, G_s)$.

Algorithm 3

Input: A precedence graph G , a system topology H , and a tardiness threshold T_{max}

Output: The minimum number of layers for a feasible schedule of G on H

```

1: for all  $0 \leq t, i_1, \dots, i_C \leq n$  do
2:    $M[t, i_1, \dots, i_C] \leftarrow -1$ 
3: return  $L(0, G)$ 

```

Procedure $L(t, G_s)$

```

1: if  $M[t, i_1, \dots, i_C] \geq 0$  then
2:   return  $M[t, i_1, \dots, i_C]$ 
3: if  $G_s = \phi$  then
4:    $M[t, i_1, \dots, i_C] \leftarrow 0$ 
5: else if  $A(G_p) = \phi$  or  $\max_{u \in \hat{U}} \{t + e(u) - d(u)\} > T_{max}$  then
6:    $M[t, i_1, \dots, i_C] \leftarrow \infty$ 
7: else
8:    $M[t, i_1, \dots, i_C] \leftarrow \infty$ 
9:   for all  $U \in A(G_p)$  do
10:     $z \leftarrow \max(L(t + 1, G_s - U), \lambda(U))$ 
11:    if  $z < M[t, i_1, \dots, i_C]$  then
12:       $M[t, i_1, \dots, i_C] \leftarrow z$ 
13: return  $M[t, i_1, \dots, i_C]$ 

```

Let C and n be the numbers of precedence chains and tasks/transactions, respectively. Algorithm 3, which implements the dynamic-programming formula in terms of recursion, maintains an entry in a $(C + 1)$ -dimensional table M for

the solution to each subproblem. Let the solution of $L(t, G_s)$ be stored in entry $M[t, i_1, i_2, \dots, i_C]$, where i_j is the number of tasks/transactions of the j_{th} partial chain of $G_p = G - G_s$. Note that each dimension needs at most $(n + 1)$ different index values because $0 \leq t, i_1, i_2, \dots, i_C \leq n$. Each table entry is initialized as -1 to indicate that the corresponding subproblem has not yet been solved. Whenever Procedure $L(t, G_s)$ is invoked, the procedure simply returns the previously derived solution in the corresponding entry if the entry has been updated. Otherwise, the solution to the subproblem is derived based on the presented dynamic-programming formula and returned. On the other hand, once the entire table is derived, a corresponding feasible schedule S can be constructed by tracing the table according to the dynamic-programming formula: We begin with the table entry that corresponds to $L(0, G)$ and schedule, at time 0, a set $U_0 \in A(\phi)$ that minimizes the solution stored in the table entry. We then compute the sets to be scheduled at the successive times recursively. In other words, we schedule, at time t , a set $U_t \in A(\cup_{i=0}^{t-1} U_i)$ that minimizes the solution in the table entry for $L(t, G - \cup_{i=0}^{t-1} U_i)$. Because we have at most n sets to determine, and each of them takes constant time $O(1)$ in making a decision (refer to the following time complexity analysis), the construction of a schedule S based on M can be done in $O(n)$ time.

5.3.2 Properties

For the rest of this section, we shall analyze the time complexity of Algorithm 3 and prove its optimality in scheduling unit tasks and unit transactions on any system topology with only one multi-layer bus.

Lemma 5.2 *The time complexity of Algorithm 3 is $O(n^{C+1})$, where C is the number of chains of a given precedence graph.*

Proof. The time complexity of the implementation depends on the number of table entries and the time required to derive the solution to a subproblem. Since each dimension of the $(C + 1)$ -dimensional table contains $n + 1$ possible index values, the table has $O(n^{C+1})$ entries. Each entry is initialized and stored with the solution to the corresponding subproblem only once (if it happens). The derivation to the solution of each subproblem refers to all of the entries that corresponds to the sets in $A(G_p)$. Since each set $U \in A(G_p)$ contains at most C tasks/transactions, $A(G_p)$ contains at most 2^C possible sets, *i.e.*, $|A(G_p)| \leq 2^C$. The verification of the validity of a set with respect to G_p (*i.e.*, in $A(G_p)$ or not) takes $O(C)$ time. For a set $U \in A(G_p)$, counting the number of transactions via bus b , *i.e.*, $\lambda(U)$, takes $O(C)$ time as well. Moreover, it takes $O(C)$ time to verify whether the scheduling of any task in \hat{U} at t fails to meet the tardiness threshold. Thus, the solution derivation of any subproblem takes $O(C + C \times 2^C) = O(1)$ time. In summary, table M can be constructed in $O(n^{C+1})$ time. ■

Theorem 5.2 *Algorithm 3 solves the tardiness-bounded layer minimization problem, when there is only one multi-layer bus, and every task/transaction is of unit execution/communication time.*

Proof. This theorem follows directly from the correctness of the dynamic-programming formula: $L(t, G_s)$. The correctness of the formula is proved by an induction on the number of tasks/transactions in G_s .

As the induction basis, the null graph does not require any bus layer regardless of when it is scheduled. Thus, $L(t, \phi) = 0$, for any $t \geq 0$. For the induction hypothesis, suppose that the formula is always correct for a suffix with at most $(n-1)$ tasks/transactions. We shall show that, for any suffix G_s with n tasks/transactions, the formula is also correct. For any suffix G_s to be scheduled from some time t , let G_p be its complementary prefix, and \hat{U} be the set of all the tasks/transactions available with respect to G_p . If $A(G_p) = \phi$ or $\max_{u \in \hat{U}} \{t + e(u) - d(u)\} > T_{max}$, then there is no feasible schedule with G_s scheduled from t . Thus, the number of layers needed is considered as ∞ . Otherwise, let U be any set in $A(G_p)$. Since U is valid with respect to G_p , its tasks/transactions can be scheduled at the same time, and the number of layers needed depends on the transactions that are via the multi-layer bus b . All the tasks/transactions of U must finish by time $t + 1$ (because they are of unit times), so the rest of G_s , *i.e.*, $(G_s - U)$, can be scheduled from $t + 1$. Since $(G_s - U)$ contains at most $n - 1$ tasks/transactions, by the induction hypothesis, $L(t + 1, G_s - U)$ is the minimum number of layers for scheduling $(G_s - U)$ from $t + 1$. Therefore, the minimum number of layers needed with U scheduled at time t is the maximum one of the two numbers. Since $A(G_p)$ contains all of the sets U valid with respect to G_p , we have $L(t, G_s) = \min_{U \in A(G_p)} \max(L(t + 1, G_s - U), \lambda(U))$. ■

5.4 Extension of the Basic Algorithm

5.4.1 Arbitrary Execution and Communication Time

This section is meant to extend Algorithm 3 to derive an optimal solution to any instance of the tardiness-bounded layer minimization problem, where tasks/transactions may have arbitrary execution/communication time. We will further extend the re-

sults of this section to the problem with more than one multi-layer buses in a later section. Note that the tardiness-bounded layer minimization problem is shown to be \mathcal{NP} -hard. The extended version of Algorithm 3 is a pseudo-polynomial time algorithm. We shall show that the time complexity of the algorithm is a polynomial time function of the total execution time and communication delay.

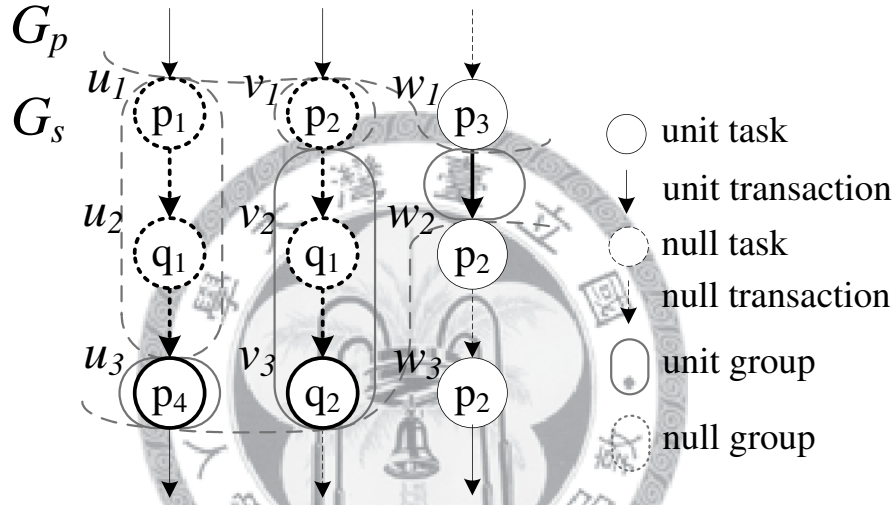


Figure 5.4: Splitting and grouping of a given precedence graph

An extended version of Algorithm 3 runs as follows: Given a precedence graph of n tasks/transactions, we first need to transform the graph into a precedence graph such that every task/transaction has an execution/communication time no more than one. Every task that has the execution time $e(u)$ is split into $e(u)$ unit tasks, and the deadline of the i_{th} unit task is set as $d(u) - e(u) + i, \forall 1 \leq i \leq e(u)$. Every transaction (u, v) of the communication delay $f(u, v)$ is split into $f(u, v)$ unit transactions. We then insert null tasks/transactions into split transactions/tasks so that tasks are separated by transactions in the resulted precedence graph. Because of the existence of null tasks/transactions (either innate or inserted), a sequence of null tasks and/or transactions may be available, at the same time, with their succeeding unit task or transaction respect to some prefix because null tasks and transactions could be done “simultaneously” and in no time (such as null tasks/transactions u_1 ,

(u_1, u_2) , u_2 , and (u_2, u_3) , and their succeeding unit task u_3 in Figure 5.4). Moreover, because a bridge task and its preceding and succeeding transactions must run without any interruption (*i.e.*, Condition 4 in Section 5.1.2), a unit task/transaction may have to run with its preceding null tasks/transactions consecutively (and without any interruption) in the same time unit (such as null tasks/transactions (v_1, v_2) , v_2 , and (v_2, v_3) , and the unit bridge task v_3 in Figure 5.4). Let a unit task/transaction and the longest sequence of null tasks/transactions that must run consecutively with the task/transaction in the same time unit be referred to as a *unit group*. Each sequence of null tasks/transactions between unit groups (if exists) is called a *null group*, and the time required to run a null group and a unit group are 0 and 1, respectively. The deadline of every group is set as the minimum deadlines of its tasks (or ∞ if no task is included). The number of layers needed by each group is set as either 0 or 1, depending on whether it includes a unit transaction that need to be transferred via bus b . Let $A^*(G_p)$ be the maximum subset of $A(G_p)$, where every set $U^* \in A^*(G_p)$ consists of some null and/or unit groups. To be more specific, if a task or transaction of some group is in $U^* \in A^*(G_p)$, then any other task or transaction of the group must also be in U^* . The extended algorithm is Algorithm 3 in which $A(G_p)$ is replaced with $A^*(G_p)$, and a group is considered as a task/transaction during the execution of the algorithm.

Lemma 5.3 *The time complexity of the extended algorithm is $O(\ell^{C+1})$, where ℓ is the total execution time and communication delay of the n tasks/transactions of a given precedence graph.*

Proof. We can assume that $n \leq \ell$; otherwise, the case is polynomial-time solvable. We shall show that the algorithm can be implemented in $O(\ell^{C+1})$ time. First, it takes

$O(\ell)$ time to split and group a given precedence graph into $O(\ell)$ null and unit groups. The time complexity of the extended algorithm depends on the number of table entries and the time required to derive a subproblem solution. Since each dimension of the table has $O(\ell)$ index values, there are $O(\ell^{C+1})$ table entries. It remains to show that a subproblem $L(t, G_s)$ can be solved in $O(1)$ time. A set $U^* \in A^*(G_p)$ can contain at most two groups of each precedence chain. It implies that there are at most 3^C possible combinations, *i.e.*, $|A^*(G_p)| \leq 3^C$. Thus, the computation of a subproblem refers to at most 3^C entries, each of which corresponds to a set $U^* \in A^*(G_p)$. Because a set U^* contains at most $2C$ groups, the corresponding verification of $U^* \in A^*(G_p)$ and the counting for $\lambda(U^*)$ can both be done in $O(C)$ time. Moreover, it takes $O(C)$ time to verify whether the scheduling of any group in \hat{U} at t fails to meet the tardiness threshold. Thus, the derivation of a subproblem solution takes $O(C + C \times 3^C) = O(1)$ time, where C is the number of chains and a constant. ■

Theorem 5.3 *The extended algorithm solves the tardiness-bounded layer minimization problem with one multi-layer bus.*

Proof. This theorem can be proved by a mathematical induction in a way similar to the proof of Theorem 5.2, if all of the sets *i.e.*, $\forall U \in A(G_p)$, are considered during the algorithm execution. We shall show that the consideration of the subset $A^*(G_p) \subseteq A(G_p)$ is sufficient for the derivation of an optimal solution. That is, for any set $U \in A(G_p)$, there exists a set $U^* \in A^*(G_p)$ such that $\max(L(t+1, G_s - U^*), \lambda(U^*)) \leq \max(L(t+1, G_s - U), \lambda(U))$.

For any set $U \in A(G_p)$, let us pick a set $U^* \in A^*(G_p)$ such that U^* is not only a superset of U but also contains the same unit tasks/transactions as U does.

Note that such a set U^* must exist, because every task/transaction of U must be in some group, and $A^*(G_p)$ contains all possible combinations of groups (available with respect to G_p). Since U^* is a superset of U , the layers required to schedule $(G_s - U^*)$ from some time $t + 1$ is clearly no more than that required to schedule $(G_s - U)$, *i.e.*, $L(t + 1, G_s - U^*) \leq L(t + 1, G_s - U)$. Moreover, U^* contains the same unit tasks/transactions as U does. It implies that the layers required to schedule all of the tasks/transactions of U^* at time t equals to that required to schedule all of the tasks/transactions of U , *i.e.*, $\lambda(U^*) = \lambda(U)$. ■

5.4.2 Multiple Multi-Layer Buses and Non-Preemptive Tasks/Transactions

The purpose of this section is to further extend the algorithm presented in Section 5.4.1 to cases with a constant number of multi-layer buses and/or non-preemptive tasks/transactions.

Considering a system topology with K multi-layer buses, let the buses be denoted as b_1, b_2, \dots , and b_K . Given a precedence graph of C precedence chains, it is not possible to have more than C transactions that are active at the same time. In other words, no bus in any optimal solution can be associated with C layers. We can simply fix the number of layers for every bus, except b_1 , and run the algorithm to minimize the number of layers required by b_1 . With such an approach, we can find out the best combination with the minimum cost by running the algorithm with different layer combinations of the other $K - 1$ buses. The time complexity is $O(C^{K-1} \times \ell^{C+1}) = O(\ell^{C+1})$, when K is a constant.

In Section 5.4.1, tasks and transactions are preemptive. We shall show

how to extend the algorithm when tasks and/or transactions are non-preemptive in their execution: Suppose that some transactions or tasks are non-preemptive. A set U available with respect to G_p is valid if the four conditions defined in Section 5.1.2 are satisfied, and all of the unit tasks/transactions spilt from an original task/transaction that is non-preemptive must run without any interruption. With the revised definition of the sets in $A(G_p)$, the optimality of the algorithm is provided for the support of the non-preemption model, and the proof of the optimality can be done in a similar way. Note that the introduction of an additional condition does not increase the number of possible sets valid with respect to a prefix, and the time complexity remains as $O(\ell^{C+1})$.

5.4.3 Different Timing Constraints and Objective Functions

Algorithm 3 and its extensions in the previous sections are meant to solve the tardiness-bounded layer minimization Problem, which is to find a feasible schedule with the maximum tardiness of tasks is bounded by a given threshold T_{max} such that the bus cost is minimized. In this section, we shall show how to extend Algorithm 3 when different timing constraints and/or objective functions are considered. We use two examples to illustrate the way for further extensions:

Suppose that we intend to find a feasible schedule with the makespan is bounded by a given threshold t_{max} such that the bus cost is minimized. The dynamic-programming formula presented in Section 5.3.1 for Algorithm 3 can be

revised as follows:

$$L(t, G_s) = \begin{cases} 0, & \text{if } G_s = \phi; \\ \infty, & \text{else if } A(G_p) = \phi \text{ or } t > t_{max}; \\ \min_{\forall U \in A(G_p)} \max(L(t+1, G_s - U), \lambda(U)), & \text{otherwise.} \end{cases}$$

Note that the only difference between this formula and that in Section 5.3.1 is the consideration of the number of needed layers as ∞ if a suffix G_s is to be scheduled from some time $t > t_{max}$.

Suppose that the number of layers of every bus of a given system topology is fixed. When we intend to find a valid schedule such that the maximum tardiness of tasks is minimized, the formula presented in Section 5.3.1 can be revised as follows:

$$T(t, G_s) = \begin{cases} 0, & \text{if } G_s = \phi; \\ \infty, & \text{else if } A(G_p) = \phi, \\ \min_{\forall U \in A(G_p)} \max(T(t+1, G_s - U), \max_{\forall u \in U} \{t + e(u) - d(u)\}), & \text{otherwise.} \end{cases}$$

Let $T(t, G_s)$ be the maximum tardiness of tasks by scheduling a suffix G_s from time t (when its complementary prefix G_p is already scheduled). Note that $T(t, \phi) = 0$, $\forall t \geq 0$. If $A(G_p) = \phi$, then no set is valid with respect to G_p , and the tardiness should be considered as ∞ . Otherwise, the maximum tardiness of a schedule, where some set $U \in A(G_p)$ is scheduled at t and $(G_s - U)$ is scheduled from $t + 1$, can be derived by comparing the tardiness of their tasks. By considering all of the possible sets $U \in A(G_p)$, the best is selected as $T(t, G_s)$. The optimality of these two extensions can be proved in a similar way, and they also have implementations with $O(\ell^{C+1})$ time complexity.

5.5 Performance Evaluation

5.5.1 Experimental Setups and Performance Metrics

The purpose of this section is to evaluate the proposed algorithm, denoted as *OPT*, in comparison with another heuristics-based approach so that the motivation and insights in the adoption of an optimal algorithm are better identified. A list-scheduling algorithm with the consideration of communication contention was revised and adopted for performance evaluation [75], denoted as *LSA*. Note that a list-scheduling algorithm was adopted because list scheduling was widely adopted in the scheduling of a precedence graph over a target system topology [55, 74], where tasks are sorted according to some priority scheme and assigned to an appropriate processor one by one in the order of their priorities. *Least Laxity First* was used as a priority scheme, because of its popularity in related study [49]. The major performance metric was the bus cost of derived topology configurations, provided a given tardiness threshold T_{max} was not violated. The impacts of different T_{max} values were also explored in the performance evaluation. The default T_{max} value of the experiments was set as 0.

An AMBA-based system topology with 9 processing elements and one shared memory unit was considered in the experiments. There were three multi-layer buses interconnected by a bridge, as shown in Figure 5.1. The *Advanced High-Performance Bus* (AHB) was adopted to connect the shared memory unit, *i.e.*, r , and three processors, *i.e.*, p_1 , p_2 , and p_3 , to provide good support to the shared memory bandwidth. The *Advanced System Bus* (ASB) served as a secondary system bus for three processors p_4 , p_5 , and p_6 , and the *Advanced Peripheral Bus* (APB) provided a communication interface for three peripheral devices p_7 , p_8 , and p_9 . Because

a high-performance bus is usually more expensive than a low-performance bus, the ratio of the costs of the three buses was set as $3^\kappa : 2^\kappa : 1^\kappa$, where κ is referred to as the *weight factor*, and the impacts of different κ values were reported. The default κ value was set as 2. Note that the AHB dominated the total cost when the κ value was large. To provide a baseline for the experimental results with different κ values, the results were normalized by $3^\kappa + 2^\kappa + 1^\kappa$.

The precedence graphs under investigation were generated by TGFF [24, 80]. Every generated graph consisted of four precedence chains, and they consisted of 100-150 tasks/transactions. Each task was pre-assigned to a processing element/memory/bridge and associated with an execution time randomly selected in the range between 1 and 3. Each transaction was associated with a bus, and its associated communication delay was randomly selected in the range between 1 and 3. The deadline of each task v was set as $d(v) = d(u) + f(u, v) + e(v) + \delta$, where u denoted the preceding task of v , and δ was an integer randomly selected between 0 and some *slack time* Δ . The impacts of different Δ values were explored in the performance evaluation. The default Δ value was set as 2. The random number generator was provided by TGFF. To emulate more realistic workloads of an AMBA-based system, TGFF was requested to generate workloads in which some certain percentage of them were associated with the highest-performance bus AHB, denoted as W_{AHB} . Different W_{AHB} values were explored to show the impacts of local bus contention.

All of the experiments were run on a platform with dual Intel[®] Xeon Quad-Core 2.5GHz CPUs and 24GB 667MHz DDR2 RAM, and the results were derived as an average value over 100 independent experiments.

5.5.2 Experimental Results

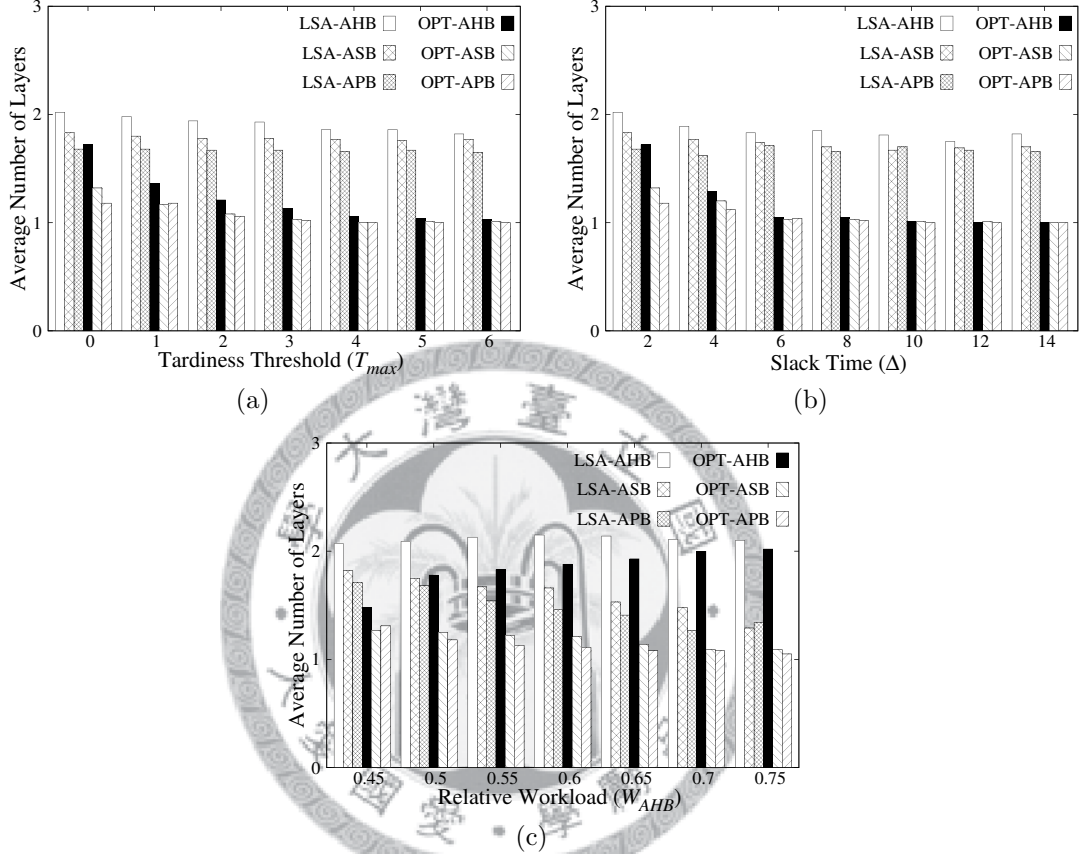


Figure 5.5: Average number of layers needed by each bus

Figure 5.5 shows the numbers of layers needed by each of the three buses under LSA (*i.e.*, LSA-AHB, LSA-ASB, and LSA-APB) and under OPT (*i.e.*, OPT-AHB, OPT-ASB, and OPT-APB) with respect to different tardiness thresholds, slack times, and relative workloads over AHB, where the preemptive task/transaction model was considered. As shown in Figure 5.5(a), the number of needed layers decreased as the given tardiness threshold increased. Nevertheless, it is worth mentioning that LSA did not decrease so dramatically as OPT did. It was because OPT always optimized the layer requirement, while LSA was likely to get unnecessary layers when it scheduled transactions heuristically. The results show that LSA could use over 1.3 times of the number of layers for each bus than OPT did when

$0 \leq T_{max} \leq 6$. As expected, the number of needed layers decreased, in general, as the deadline was relaxed, as shown in Figure 5.5(b). We observed that the number of needed layers determined by LSA did not decrease monotonously as the deadline was relaxed. The reason behind the observation was that LSA was a greedy algorithm that tended to postpone task executions to the future so as to minimize the number of needed layers for the present time. Some tasks might become very urgent in execution unexpectedly and, consequently, the number of needed layers might not absolutely depend on the relaxing degree of the deadline. The results show that LSA could use over 1.3 times of the number of layers than OPT did when $2 \leq \Delta \leq 14$. As shown in Figure 5.5(c), the number of layers needed by AHB increased while those needed by ASB and APB decreased, in general, as the relative workload over AHB increased. We must point out that the number of needed layers determined by OPT changed significantly with the changing of the workloads, and it means that the buses were heavily congested. OPT was very efficient in bus usage. The results show that LSA could use over 1.2 times of the number of layers than OPT did when $45\% \leq W_{AHB} \leq 75\%$.

Figure 5.6 shows the impacts of the relative weights of buses on the bus cost under the preemptive task/transaction model. As we can see in Figures 5.6(a), 5.6(b), and 5.6(c), the bus cost increased as the relative weight of AHB increased. It was mainly resulted from the fact that AHB usually needed more layers than other buses (due to the heavy workload), and AHB would dominate the bus cost when it was of a large relative weight. Moreover, the degree of the increasing depended on the difference in the numbers of needed layers between AHB and other buses (refer to Figure 5.5). It was also the reason why the bus cost achieved different degrees of increasing values with different tardiness thresholds (/slack times/relative workloads). Figure 5.6(a) shows that, when $T_{max} = 0$ and κ ranged from 0 to 5, LSA

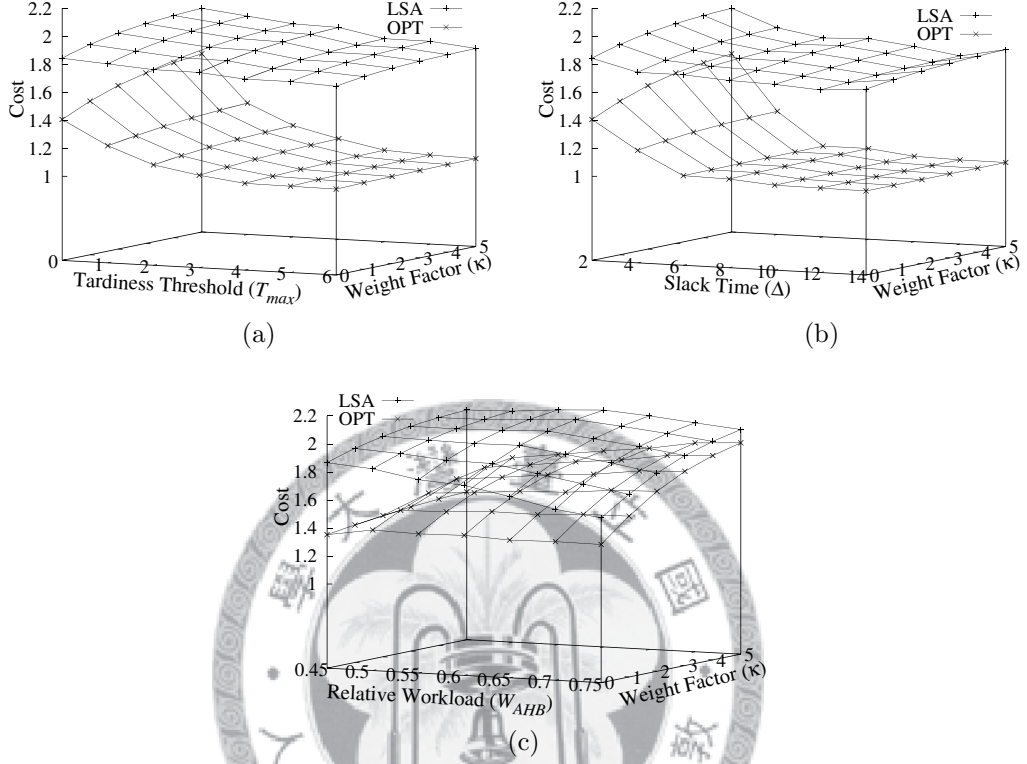


Figure 5.6: Impacts of the relative weight of buses

could spend additional cost by 31% to 19% to satisfy a given tardiness threshold, compared with an optimal topology configuration. When $\Delta = 2$, Figure 5.6(b) shows similar results because the two experiments shared the same setting (*i.e.*, $T_{max} = 0$ and $\Delta = 2$). Figure 5.6(c) shows that LSA could spend additional cost by 30% to 23%, when $W_{AHB} \approx 45\%$ and κ ranged from 0 to 5. Note that the additional cost reduced as κ increased, because the difference in the number of needed layers determined by OPT for each bus in those cases was more notable.

Figure 5.7 shows the bus costs of topology configurations derived by LSA and OPT, when the preemptive (*i.e.*, LSA-P and OPT-P) and non-preemptive (*i.e.*, LSA-NP and OPT-NP) task/transaction models were considered, respectively. As can be seen in Figures 5.7(a), 5.7(b), and 5.7(c), the bus cost under the non-preemptive task/transaction model was higher than that when the preemptive one

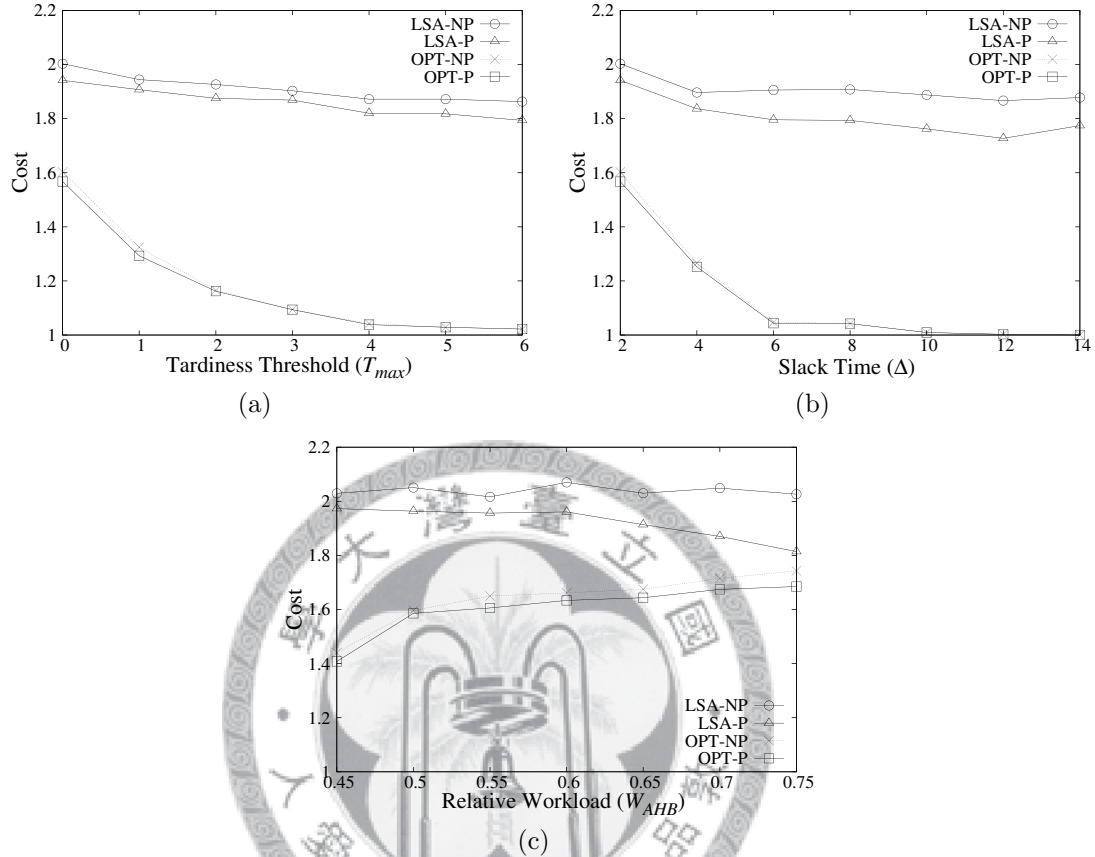


Figure 5.7: Preemptive task/transaction model vs. non-preemptive task/transaction model

was considered. Such a phenomenon was expectable, because a valid schedule under the non-preemptive task/transaction model was also valid under the preemptive one, and not vice versa. Moreover, LSA was more sensitive to the non-preemptive task/transaction model than OPT did. The reason was that the tasks/transactions spilt from any non-preemptive task/transaction had to run without any interruption. It was hard for a greedy algorithm to make a “good” decision. It was observed that LSA was even more sensitive when tasks/transactions were of longer execution times/communication delays. Figures 5.7(a), 5.7(b), and 5.7(c) show that the non-preemptive task/transaction model could cause OPT (/LSA) to increase the bus cost up to 2.3% (/3.9%), 2.3% (/8.1%), and 2.8% (/9.6%) respectively, compared to that under the preemptive one. In Figure 5.7(c), the reason for the reducing in the

bus cost when $60\% \leq W_{AHB} \leq 75\%$ under the preemptive task/transaction model was that the cost reducing for ASB and APB was more significant than the cost increasing for AHB when the workloads over on ASB and APB were low (refer to Figure 5.5(c)).

Algorithm	T_{max}	Δ	W_{AHB}
LSA	4.52×10^{-4}	4.01×10^{-4}	3.29×10^{-4}
OPT	1.63	14.60	2.24

Table 5.1: Average running time (second) required per graph

Table 5.1 shows the average running time required to derive a topology configuration for a precedence graph in the experiments with respect to the tardiness threshold (T_{max}), slack time (Δ), and relative workload (W_{AHB}), respectively, under the preemptive task/transaction model. Note that the time required for data input/output was excluded from the results. The results show that LSA was much more efficient than OPT in terms of the running time. Solutions could often be resolved by LSA in a millisecond, and different parameter settings had no significant impacts on its running time. In contrast, OPT took more time to resolve a problem instance in the experiments with respect to the slack time than those in the other two experiments. The reason was that the search space of solutions apparently increased as the deadline was relaxed (because fewer branches in the *recursion tree* were trimmed, due to deadline misses). The average run time of OPT ranged from 1.63 seconds to 14.60 seconds with respect to different experiment settings. In terms of system designs, the running time was considered short. Moreover, the search space would not unlimitedly increase (once no solution is trimmed from the recursion tree). We observed that OPT took 24 minutes to search the whole recursion tree (when the deadlines of tasks were set as ∞). It implies that there were lost of unfeasible solutions trimmed by OPT in the recursion tree (due to deadline misses). We also observed that OPT was more efficient when the non-preemptive

task/transaction model was considered. It was because the solution space under this model was a subset of that under the preemptive one.

5.6 Summary

This paper proposes a dynamic-programming approach for scheduling real-time tasks with chain-based precedence constraints over multi-layer bus systems. We first show the \mathcal{NP} -hardness of the target problem and the non-existence of any approximation algorithm with ratio better than 1.5, unless $\mathcal{P} = \mathcal{NP}$. As opposed to heuristic approaches, the proposed approach is proved to be optimal in the provision of performance guarantees with the minimum cost, and its time complexity is shown to be pseudo-polynomial in the total execution time and communication delay. In addition to the results, we show how to revise the proposed approach for problems with different timing constraints and/or objective functions. The capability of the proposed approach was evaluated over an AMBA-based system topology [53] with tasks generated by TGFF [80], so as to better identify the motivation and insights in the adoption of an optimal algorithm. The experimental results show that the proposed approach is very effective in the bus cost minimization, and its running time only ranges from seconds to minutes for the experimental settings. The rationale behind the efficient performance is that unfeasible schedules often dominate the space of possible solutions and are excluded by the proposed approach.

Chapter 6

Concluding Remarks

6.1 Conclusion

This dissertation explores energy and cost efficiency in data communication. It is dedicated to the provision of not only algorithms with guaranteed optimality but also fundamental negative results for the target problems. It also provides extensive studies on the performance evaluation of the proposed algorithms with a series of experiments. With not only theoretical analysis, the proposed algorithms are also shown to be applicable and practical in the optimization of energy consumption and cost.

In the first part of this dissertation, we focus on energy-efficient routing for inter-device data communication. This part studies power-aware routing so as to maximize the minimum remaining energy of all nodes after routing. The routing metric, referred to as Maximum-Residual Routing, aims at maintaining the minimum remaining energy as high as possible so as to delay the first failure time of nodes

in the network. In this part, a polynomial-time optimal algorithm is proposed for Maximum-Residual Multicasting when up-to-date topology and energy information are available. We then show that Maximum-Residual Aggregating is \mathcal{NP} -hard and that, unless $\mathcal{P} = \mathcal{NP}$, its minimization version cannot be approximated within a ratio of $(2 - \epsilon)$ for any $\epsilon > 0$. The proposed routing algorithm for Maximum-Residual Multicasting can be applied to existing routing protocols, especially those based on the link-state approach. We have conducted extensive simulations to better understand the properties of the routing metric. Based on the experiments, we provide some interesting observations and conclude that using the routing metric to find routes is very beneficial because (1) network lifetime is significantly improved and (2) variance in remaining energy is significantly reduced, compared with other work [50, 72, 77, 85].

The second part of this dissertation continues the study of the first part and considers applications with a huge population of mobile devices such that no global information can be efficiently maintained at any node. We first propose a distributed algorithm for Maximum-Residual Multicast and prove its optimality without the considerations of node movements and control overheads. When mobility and control message collisions are taken into consideration, it is shown that every derived route remains loop-free and converges toward an optimal solution in the maximization of the minimum residual energy. Based on the proposed algorithm, we then develop a routing protocol, namely the Maximum-Residual Multicast Protocol (MRMP), which is adaptable to network topologies and resources that may change over time. MRMP is a source-initiated on-demand multicast protocol, where no periodic control message is needed in the information collection of the network topology and remaining energy. Each multicast tree is derived based on the individual decisions of intermediate nodes and the decisions form a loop-free

multicast tree naturally. MRMP has demonstrated itself being effective and efficient in power-aware routing over NS2 based on parameter setting of Intel[®] wireless devices [2] and performance metrics concerned by the IETF MANET Working Group [22]. The proposed distributed methodology is also applicable to various related optimization problems (such as the minimization of the total energy consumption of any path from a source to a destination) and provides useful insights when network resources (such as bandwidth) might change over time.

In the third part of this dissertation, we focus on cost-efficient scheduling for intra-device data communication. This part explores real-time task scheduling with chain-based precedence constraints in embedded systems with multi-layer buses. The objective is to minimize the total cost contributed by the needed bus layers without any violation of timing constraints. The contributions of this work start with fundamental but negative results on the \mathcal{NP} -hardness of the target problem and its inapproximability ratio. To be more specific, we show that, unless $\mathcal{P} = \mathcal{NP}$, it is not possible to have any approximation algorithm with an approximation ratio better than 1.5. A polynomial-time optimal algorithm, based on dynamic programming, is then proposed for a restricted case in which one multi-layer bus, and unit execution and communication time are considered. The result is then extended as a pseudo-polynomial-time optimal algorithm in the considerations of multiple multi-layer buses, arbitrary execution and communication time, and different timing constraints and objective functions. The capability of the proposed algorithm was evaluated over an AMBA-based system topology [53] with tasks generated by TGFF [80], so as to better identify the motivation and insights in the adoption of an optimal algorithm. The experimental results show that the proposed algorithm is very effective in the bus cost minimization, and its running time only ranges from seconds to minutes for the experimental settings. The rationale behind the effi-

cient performance is that unfeasible schedules often dominate the space of possible solutions and are excluded by the proposed algorithm.

6.2 Future Research Directions

Energy- and cost-efficiency in data communication have a wide spectrum and might involve with multiple engineering disciplines. In the near future, we shall explore the problems that are closest to those studied in this dissertation and remain open. Future research directions might also consider the implementation and application of the proposed methodologies to existing systems/tools. More research on networked embedded systems with the considerations of energy- and cost-efficiency might prove very rewarding in the future.

6.2.1 Residual-Energy Maximization

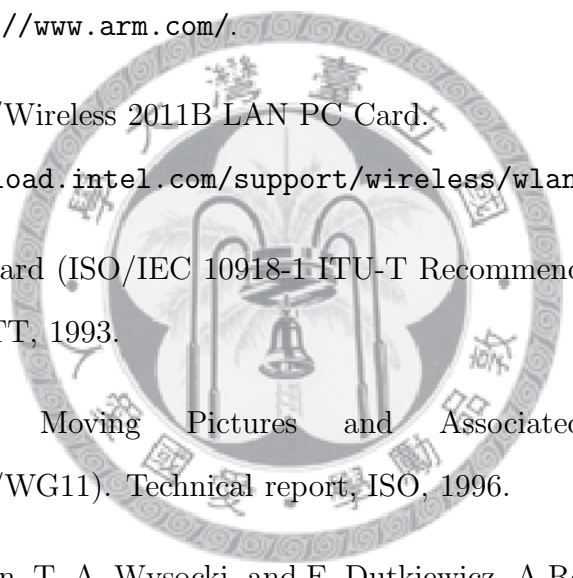
We shall further explore whether the techniques used in more efficient minimum-spanning-tree algorithms can be applied to Maximum-Residual Multicasting. We shall also exploit approximation algorithms for the minimization version of Maximum-Residual Aggregating with an objective to approach the ratio 2 and identify the asymptotic hardness of the geometric version. For another direction, we shall explore the *multi-source maximum-residual multicast problem*, where multiple sources are considered simultaneously. Note that such a problem is NP-hard even with global topology and energy information. Good studies over different mobility patterns, such as Random-Waypoint, Gauss-Markov, and City Section Mobility Models [11], might also help a lot in power-aware routing problems.

6.2.2 Bus-Layer Minimization

We shall further extend the work by considering more complicated precedence constraints that could cover a wider spectrum of applications. We shall also apply the proposed methodology to existing electronic design automation (EDA) tools, such as [52], in the optimization of the system cost at an early stage in system designs.



Bibliography

- 
- [1] ARM. <http://www.arm.com/>.
- [2] Intel® PRO/Wireless 2011B LAN PC Card.
<ftp://download.intel.com/support/wireless/wlan/pro2011/wireless.pdf>.
- [3] JPEG Standard (ISO/IEC 10918-1 ITU-T Recommendation T.81). Technical report, CCITT, 1993.
- [4] Coding of Moving Pictures and Associated Audio (ISO/IEC JTC1/SC29/WG11). Technical report, ISO, 1996.
- [5] M. Abolhasan, T. A. Wysocki, and E. Dutkiewicz. A Review of Routing Protocols for Mobile Ad Hoc Networks. In *Ad Hoc Networks*, volume 2, pages 1–22, 2004.
- [6] D. P. Agrawal and Q.-A. Zeng. *Introduction to Wireless and Mobile Systems*, pages 59–77, 297–300. Thomson Brooks/Cole, 2003.
- [7] C. Ambühl. An Optimal Bound for the MST Algorithm to Compute Energy Efficient Broadcast Trees in Wireless Networks. In *Proc. of the ICALP*, pages 1139–1150, 2005.
- [8] D. Bertsekas and R. Gallager. *Data Networks*, pages 325–333. Prentice Hall, Inc., 1987.

- [9] C. Brandolese. Source-Level Estimation of Energy Consumption and Execution Time of Embedded Software. In *Proc. of the DSD*, pages 115–123, 2008.
- [10] C. Buragohain, D. Agrawal, and S. Suri. Power Aware Routing for Sensor Databases. In *Proc. of the IEEE INFOCOM*, pages 1747–1757, 2005.
- [11] T. Camp, J. Boleng, and V. Davies. A Survey of Mobility Models for Ad Hoc Network Research. In *WCMC: Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications*, volume 2, pages 483–502, 2002.
- [12] I. Caragiannis, M. Flammini, and L. Moscardelli. An Exponential Improvement on the MST Heuristic for Minimum Energy Broadcasting in Ad Hoc Wireless Networks. In *Proc. of the ICALP*, pages 447–458, 2007.
- [13] G. Călinescu, S. Kapoor, A. Olshevsky, and A. Zelikovsky. Network Lifetime and Power Assignment in Ad-Hoc Wireless Networks. In *Proc. of the ESA*, pages 114–126, 2003.
- [14] J. Chang and L. Tassiulas. Maximum Lifetime Routing in Wireless Sensor Networks. In *IEEE/ACM Transactions on Networking*, volume 12, pages 609–619, 2004.
- [15] J.-H. Chang and L. Tassiulas. Energy Conserving Routing in Wireless Ad-hoc Networks. In *Proc. of the IEEE INFOCOM*, pages 22–31, 2000.
- [16] J. J. Chang, P. C. Hsiu, and T. W. Kuo. Search-Oriented Deployment Strategies for Wireless Sensor Networks. In *Proc. of the IEEE ISORC*, pages 164–171, 2007.
- [17] M. Čagalj, J. Hubaux, and C. Enz. Minimum-Energy Broadcast in All-Wireless Networks: NP-Completeness and Distribution Issues. In *Proc. of the ACM MobiCom*, pages 172–182, 2002.

- [18] C.-C. Chiang, M. Gerla, , and L. Zhang. Forwarding Group Multicast Protocol (FGMP) for Multihop, Mobile Wireless Networks. In *ACM/Baltzer Journal of Cluster Computing*, volume 1, pages 187–196, 1998.
- [19] P. Chrétienne and C. Picouleau. *Scheduling with Communication Delays: A Survey*, pages 65–90. John Wiley & Sons Ltd, 1995.
- [20] A. E. F. Clementi, P. Crescenzi, P. Penna, G. Rossi, and P. Vocca. On the Complexity of Computing Minimum Energy Consumption Broadcast Subgraphs. In *Proc. of STACS*, pages 121–131, 2001.
- [21] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*, pages 532–535, 561–579. MIT Press, 1990.
- [22] S. Corson and J. Macker. Mobile Ad Hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations. 1999. RFC 2501.
- [23] C. de M. Cordeiro, H. Gossain, and D. P. Agrawal. Multicast over Wireless Mobile Ad Hoc Networks: Present and Future Directions. In *IEEE Network*, volume 17, pages 52–59, 2003.
- [24] R. P. Dick, D. L. Rhodes, and W. Wolf. TGFF: Task Graphs for Free. In *Proc. of the IEEE CODES/CASHE*, pages 97–101, 1998.
- [25] D.W. Engels, J. Feldman, D.R. Karger, and M.Ruhl. Parallel Processor Scheduling with Delay Constraints. In *Proc. of the ACM-SIAM SODA*, pages 577–585, 2001.
- [26] K. Fall and K. Varadhan. The ns Manual. Technical report, UC Berkeley and LBL and USC/ISI and Xerox PARC, 2007.

- [27] J. J. Garcia-Luna-Aceves and E. L. Madruga. The Core-Assisted Mesh Protocol. In *IEEE Journal on Selected Areas in Communications*, volume 17, pages 1380–1394, 1999.
- [28] J. J. Garcia-Luna-Aceves and C. M. Spohn. Source-Tree Routing in Wireless Networks. In *Proc. of the IEEE ICNP*, pages 273–282, 1999.
- [29] M. Garey and D. Johnson. *Computers and Intractability: A guide to the Theory of NP-Completeness*. Freeman, 1979.
- [30] Z. Genc and O. Ozkasap. EraMobile: Epidemic-Based Reliable and Adaptive Multicast for MANETs. In *Proc. of the IEEE WCNC*, pages 4395–4400, 2007.
- [31] IEEE 802.11 Working Group. Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. In *ANSI/IEEE Std 802.11*, 1999.
- [32] Frank Harary. *Graph Theory*, page 172. Addison-Wesley, 1969.
- [33] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. In *Proc. of the IEEE HICSS*, pages 1–10, 2000.
- [34] E. Huang, W. Hu, J. Crowcroft, and I. Wassell. Towards Commercial Mobile Ad Hoc Network Applications: A Radio Dispatch System. In *Proc. of the ACM MobiHoc*, pages 355–365, 2005.
- [35] P. Jacquet, P. Muhlethaler, T. Clausen, A. Laouiti, A. Qayyum, and L. Viennot. Optimized Link State Routing Protocol for Ad Hoc Networks. In *Proc. of the IEEE INMIC*, pages 62–68, 2001.
- [36] L. Ji and M. S. Corson. A Lightweight Adaptive Multicast Algorithm. In *Proc. of the IEEE GLOBECOM*, pages 1036–1042, 1998.

- [37] L. Ji and M.S. Corson. Differential Destination Multicast - A MANET Multicast Routing Protocol for Small Groups. In *Proc. of the IEEE INFOCOM*, pages 1192–1201, 2001.
- [38] D. B. Johnson and D. A. Maltz. Dynamic Source Routing in Ad Hoc Wireless Networks. In *Mobile Computing*, pages 153–181. Kluwer Academic Publishers, 1996.
- [39] K. Kalpakis, K. Dasgupta, and P. Namjoshi. Maximum Lifetime Data Gathering and Aggregation in Wireless Sensor Networks. In *Proc. of the IEEE International Conference on Networking*, pages 685–696, 2002.
- [40] H. Karl and A. Willig. *Protocols and Architectures for Wireless Sensor Networks*, pages 232–250. John Wiley & Sons Inc., 2005.
- [41] L. Kelly, P. Piguet, and C. Piguet. *Low-Power Electronics Design*. CRC Press, 2005.
- [42] B. Krishnamachari, D. Estrin, and S. B. Wicker. The Impact of Data Aggregation in Wireless Sensor Networks. In *Proc. of ICDCS*, pages 575–578, 2002.
- [43] P. Kulasinghe and A. El-Amawy. On the Complexity of Optimal Based Interconnections. In *IEEE Trans. on Computers*, volume 44, pages 1248–1251, 1995.
- [44] T. Kunz and E. Cheng. On-Demand Multicasting in Ad-Hoc Networks: Comparing AODV and ODMRP. In *Proc. of the IEEE ICDCS*, pages 453–454, 2002.
- [45] J. F. Kurose and K. W. Ross. *Computer Networking: A Top-Down Approach Featuring the Internet*, pages 451–455, 484–486. Addison Wesley, 2nd edition, 2003.

- [46] K. Lahiri, A. Raghunathan, and S. Dey. Design Space Exploration for Optimizing On-Chip Communication Architectures. In *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, volume 23, pages 952–961, 2004.
- [47] S.-J. Lee, M. Gerla, and C.-C. Chiang. On-Demand Multicast Routing Protocol. In *Proc. of the IEEE WCNC*, pages 1298–1304, 1999.
- [48] S.-J. Lee, W. Su, J. Hsu, M. Gerla, and R. Bagrodia. A Performance Comparison Study of Ad Hoc Wireless Multicast Protocols. In *Proc. of the IEEE INFOCOM*, pages 565–574, 2000.
- [49] J.Y.-T. Leung. A New Algorithm for Scheduling Periodic, Real-Time Tasks. In *Algorithmica*, volume 4, pages 209–219, 1989.
- [50] Q. Li, J. Aslam, and D. Rus. Online Power-Aware Routing in Wireless Ad Hoc Networks. In *Proc. of the ACM MobiCom*, pages 91–107, 2001.
- [51] B. Liang and Z. J. Haas. Predictive Distance-Based Mobility Management for PCS Networks. In *Proc. of the IEEE INFOCOM*, pages 1377–1384, 1999.
- [52] C.-M. Lien, Y.-S. Chen, and C.-S. Shih. On-Chip Bus Architecture Optimization for Multi-core SoC Systems. In *Proc. of the IFIP SEUS*, pages 301–310, 2007.
- [53] ARM Limited. AMBA 2.0 Specification. Technical report, 1999.
- [54] ARM Limited. Multi-layer AHB Overview. Technical report, 2004.
- [55] G. Q. Liu, K.L. Poh, and M. Xie. Iterative List Scheduling for Heterogeneous Computing. In *Journal of Parallel and Distributed Computing*, volume 65, pages 654–665, 2005.

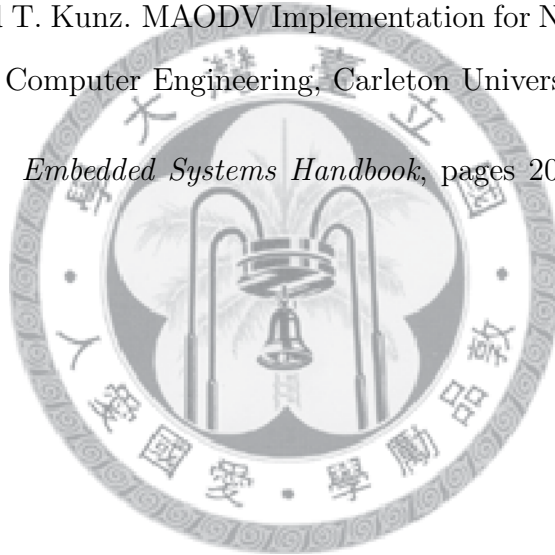
- [56] C. P. Low and L. W. Goh. On the Construction of Energy-Efficient Maximum Residual Battery Capacity Broadcast Trees in Static Ad Hoc Wireless Networks. In *Computer Communications*, volume 29, pages 93–102, 2005.
- [57] P. Markenscoff. Bus Scheduling for a Multiple-Processor System with Shared Buses. In *IEEE Computers and Digital Techniques*, volume 134, pages 288–294, 1987.
- [58] P. Mohapatra and S. Krishnamurthy. *Ad Hoc Networks: Technologies and Protocols*, pages 1–22. Springer, 2005.
- [59] R. Morris, J. Jannotti, M. F. Kaashoek, J. Li, and D. Decouto. CarNet: A Scalable Ad Hoc Wireless Network System. In *Proc. of the ACM SIGOPS*, pages 61–65, 2000.
- [60] T. N. Mudge, J. P. Hayes, and D. C. Winsor. Multiple Bus Architectures. In *IEEE Computer*, volume 20, pages 42–48, 1987.
- [61] O. Ogawa, S.B. de Noyerand P. Chauvet, K. Shinohara, Y. Watanabe, H. Nizuma, T. Sasaki, and Y. Takai. A Practical Approach for Bus Architecture Optimization at Transaction Level. In *Proc. of the IEEE/ACM DATE*, pages 176–181, 2003.
- [62] A. Orda and B.-A. Yassour. Maximum-Lifetime Routing Algorithms for Networks with Omnidirectional and Directional Antennas. In *Proc. of the ACM MobiHoc*, pages 426–437, 2005.
- [63] J. Park and S. Sahni. An Online Heuristic for Maximum Lifetime Routing in Wireless Sensor Networks. In *IEEE Transactions on Computers*, volume 55, pages 1048–1056, 2006.

- [64] C. E. Perkins and E. M. Royer. Ad-hoc On-Demand Distance Vector Routing. In *Proceedings of the IEEE WMCSA*, pages 90–100, 1999.
- [65] H. Posadas, F. Herrera, P. Sanchez, E. Villar, and F. Blasco. System-Level Performance Analysis in SystemC. In *Proc. of the IEEE/ACM DATE*, pages 378–383, 2004.
- [66] E. M. Royer and C. E. Perkins. Multicast Operation of the Ad Hoc On-Demand Distance Vector Routing Protocol. In *Proc. of the ACM Mobicom*, pages 207–218, 1999.
- [67] E. M. Royer and C. E. Perkins. Multicast Ad hoc On-Demand Distance Vector (MAODV) Routing. 2000. Internet Draft, draft-ietf-manet-maodv-00.txt, work in progress.
- [68] K.K. Ryu and V. J. Mooney III. Automated Bus Generation for Multiprocessor SoC Design. In *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, volume 23, pages 1531–1549, 2004.
- [69] A. Sankar and Z. Liu. Maximum Lifetime Routing in Wireless Ad-hoc Networks. In *Proc. of the IEEE INFOCOM*, pages 1089–1097, 2004.
- [70] M. Sheliga and E. H.-M. Sha. Bus Minimization and Scheduling of Multi-Chip Systems. In *Proc. of the ACM/IEEE GLS-VLSI*, pages 40–45, 1995.
- [71] G.C. Sih and E.A. Lee. A Compile-Time Scheduling Heuristic for Interconnection-Constrained Heterogeneous Processor Architectures. In *IEEE Trans. on Parallel and Distributed Systems*, volume 4, pages 175–187, 1993.
- [72] S. Singh, C. S. Raghavendra, and J. Stepanek. Power-Aware Broadcasting in Mobile Ad Hoc Networks. In *Proc. of the IEEE PIMRC*, pages 22–31, 1999.

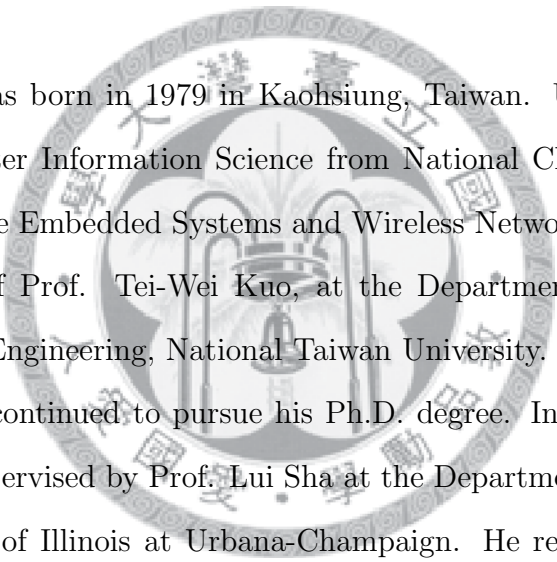
- [73] S. Singh, M. Woo, and C. S. Raghavendra. Power-Aware Routing in Mobile Ad Hoc Networks. In *Proc. of the ACM MobiCom*, pages 181–190, 1998.
- [74] O. Sinnen and L. Sousa. List Scheduling: Extension for Contention Awareness and Evaluation of Node Priorities for Heterogeneous Cluster Architectures. In *Parallel Computing*, volume 30, pages 81–101, 2004.
- [75] O. Sinnen and L.A. Sousa. Communication Contention in Task Scheduling. In *IEEE Trans. on Parallel and Distributed Systems*, volume 16, pages 503–515, 2005.
- [76] G. Tel. *Introduction to Distributed Algorithms*, pages 121–123. Cambridge University Press, 2nd edition, 2000.
- [77] C.-K. Toh. Maximum Battery Life Routing to Support Ubiquitous Mobile Computing in Wireless Ad Hoc Networks. In *IEEE Communications Magazine*, volume 39, pages 138–147, 2001.
- [78] Y. Toor, P. Muhlethaler, and A. Laouiti. Vehicle Ad Hoc Networks: Applications and Related Technical Issues. In *IEEE Communications Surveys & Tutorials*, volume 10, pages 74–88, 2008.
- [79] Y. Ueda, H. Yamauchi, M. Mukuno, S. Furuichi, M. Fujisawa, F. Qiao, and H.Z. Yang. 6.33mW MPEG Audio Decoding on a Multimedia Processor. In *Proc. of the IEEE ISSCC*, pages 1636–1645, 2006.
- [80] K. Vallerio. Task Graphs for Free (TGFF v3.0). Technical report, 2008.
- [81] T.A. Varvarigou, V.P. Roychowdhury, T. Kailath, and E.L. Lawler. Scheduling In and Out Forests in the Presence of Communication Delays. In *IEEE Trans. on Parallel and Distributed Systems*, volume 7, pages 1065–1074, 1996.

- [82] J. Verriet. The Complexity of Scheduling Graphs of Bounded Width Subject to Non-zero Communication Delays. Technical report, Utrecht University, 1997.
- [83] K. Viswanath, K. Obraczka, and G. Tsudik. Exploring Mesh- and Tree Based Multicast Routing Protocols for MANETs. In *IEEE Transactions on Mobile Computing*, volume 5, pages 28–42, 2006.
- [84] P.-J. Wan, G. Călinescu, X.-Y. Li, and O. Frieder. Minimum-Energy Broadcast Routing in Static Ad Hoc Wireless Networks. In *Proceedings of the IEEE INFOCOM*, pages 1162–1171, 2001.
- [85] J. E. Wieselthier, G. D. Nguyen, and A. Ephremides. On the Construction of Energy-Efficient Broadcast and Multicast Trees in Wireless Networks. In *Proc. of the IEEE INFOCOM*, pages 585–594, 2000.
- [86] C. W. Wu and Y.C. Tay. AMRIS: A Multicast Protocol for Ad hoc Wireless Networks. In *Proc. of the IEEE MILCOM*, pages 25–29, 1999.
- [87] J. Xie, R. R. Talpade, A. Mcauley, and R. Talpade. AMRoute: Ad Hoc Multicast Routing Protocol. In *Mobile Networks and Applications (MONET)*, volume 7, pages 429–439, 2002.
- [88] Q. Xie, C.-T. Lea, M. J. Golin, and R. Fleischer. Maximum Residual Energy Routing with Reverse Energy Cost. In *Proc. of the IEEE GLOBECOM*, pages 564–569, 2003.
- [89] T. Yang and A. Gerasoulis. DSC: Scheduling Parallel Tasks on an Unbounded Number of Processors. In *IEEE Trans. on Parallel and Distributed Systems*, volume 5, pages 951–967, 1994.

- [90] W. Ye, J. Heidemann, and D. Estrin. An Energy-Efficient MAC Protocol for Wireless Sensor Networks. In *Proc. of the IEEE INFOCOM*, pages 1567–1576, 2002.
- [91] Y. Yu, R.h Govindan, and D. Estrin. Geographical and Energy-Aware Routing: A Recursive Data Dissemination Protocol for Wireless Sensor Networks. Technical report, UCLA Computer Science Department, 2001.
- [92] Y.F. Zhu and T. Kunz. MAODV Implementation for NS-2.26. Technical report, Systems and Computer Engineering, Carleton University, 2004.
- [93] R. Zurawski. *Embedded Systems Handbook*, pages 20(1)–20(22). CRC Press, 2006.



Curriculum Vitae



Pi-Cheng Hsiu was born in 1979 in Kaohsiung, Taiwan. Upon receiving the B.S. degree in Computer Information Science from National Chiao Tung University in 2002, he joined the Embedded Systems and Wireless Networking Laboratory, under the supervision of Prof. Tei-Wei Kuo, at the Department of Computer Science and Information Engineering, National Taiwan University. In 2004, he received the M.S. degree and continued to pursue his Ph.D. degree. In 2007, he was a visiting Ph.D. student supervised by Prof. Lui Sha at the Department of Computer Science in the University of Illinois at Urbana-Champaign. He received his Ph.D. degree in Computer Science and Information Engineering in 2009. His research interests include energy-efficient routing, real-time scheduling, and mobile computing.

Publication List

Journal Papers

1. **Pi-Cheng Hsiu**, Chin-Hsien Wu, and Tei-Wei Kuo, “Maximum-Residual Multicasting and Aggregating in Wireless Ad Hoc Networks,” *accepted and to appear in ACM/Springer Wireless Networks (WINET)*.
2. **Pi-Cheng Hsiu** and Tei-Wei Kuo, “A Maximum-Residual Multicast Protocol for Large-Scale Mobile Ad Hoc Networks,” *accepted and to appear in IEEE Transactions on Mobile Computing*.

Conference Papers

1. Su-Fang Hsiao, **Pi-Cheng Hsiu**, and Tei-Wei Kuo, “A Reconfigurable Virtual Storage Device,” *in Proceedings of the IEEE International Symposium on Object and component-oriented Real-time distributed Computing (ISORC)*, pages 80-87, 2009.
2. Yu-Kai Huang, **Pi-Cheng Hsiu**, Wei-Ni Chu, Kuan-Chang Hung, Ai-Chun Pang, Tei-Wei Kuo, Min Di, and Hua-Wei Fang, “An Integrated Deployment Tool for ZigBee-based Wireless Sensor Networks,” *in Proceedings of the IEEE/IFIP International Conference on Embedded and Ubiquitous Computing*

- (*EUC*), Vol. 1, pages 309-315, 2008.
3. Jiun-Jian Chang, **Pi-Cheng Hsiu**, and Tei-Wei Kuo, "Search-Oriented Deployment Strategies for Wireless Sensor Networks," in *Proceedings of the IEEE International Symposium on Object and component-oriented Real-time distributed Computing (ISORC)*, pages 164-171, 2007.
 4. Jane W-S. Liu, Chi-Sheng Shih, Han-Chun Yeh, **Pi-Cheng Hsiu**, Wen-Hsien Chang, Pei-Hsuan Tsai, and Chin-Yen Yu, "Point-of-Care Support for Error-Free Medication Process," in *Proceedings of the IEEE Joint Joint Workshop on High Confidence Medical Devices, Software, and Systems and Medical Device Plug-and-Play Interoperability (HCMDSS-MDPnP)*, pages 34-35, 2007.
 5. Pei-Hsuan Tsai, Han-Chun Yeh, Chin-Yen Yu, **Pi-Cheng Hsiu**, Chi-Sheng Shih, and Jane W. S. Liu, "Compliance Enforcement of Temporal and Dosage Constraints," in *Proceedings of the IEEE Real-Time Systems Symposium (RTSS)*, pages 359-368, 2006.
 6. Han-Chun Yeh, **Pi-Cheng Hsiu**, Pei-Hsuan Tsai, Chi-Sheng Shih, and Jane W. S. Liu, "APAMAT: A Prescription Algebra for Medication Authoring Tool," in *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Vol. 5, pages 4284-4291, 2006.
 7. Chung-Fan Hsu, Mark H. Y. Liao, **Pi-Cheng Hsiu**, Yeu-Shian Lin, Chi-Sheng Shih, Tei-Wei Kuo, and Jane W. S. Liu, "Smart Pantries for Homes," in *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Vol. 5, pages 4276-4283, 2006.
 8. **Pi-Cheng Hsiu**, Chin-Fu Kuo, Tei-Wei Kuo, and Eric Y. T. Juan, "Scenario Based Threat Detection and Attack Analysis," in *Proceedings of the IEEE*

International Carnahan Conference on Security Technology (ICCST), pages 279-282, 2005.

Technical Reports

1. Pei-Hsuan Tsai, Han-Chun Yeh, **Pi-Cheng Hsiu**, Chi-Sheng Shi, Tei-Wei Kuo, and Jane W. S. Liu, “A Scarce Resource Model for Medication Scheduling,” No. TR-IIS-06-003, Institute of Information Science, Academia Sinica, Taipei, Taiwan, 2006.
2. **Pi-Cheng Hsiu**, Han-Chun Yeh, Pei-Hsuan Tsai, Chi-Sheng Shih, Daniel H. Burkhardt, Tei-Wei Kuo, Jane W. S. Liu, and Tai-Yi Huang, “A general model for medication scheduling,” No. TR-IIS-05-008, Institute of Information Sciences, Academia Sinica, Taipei, Taiwan, 2005.

