國立臺灣大學資訊管理學研究所碩士論文

指導教授： 林永松 博士

具資料集縮能力無線感測網路之
低能耗與延遲排程演算法

An Energy and Delay Efficient Scheduling Algorithm for Data-Centric Wireless Sensor Networks

研究生： 王弘翕 撰

中華民國九十五年七月

具資料集縮能力無線感測網路之
低能耗與延遲排程演算法

An Energy and Delay Efficient
Scheduling Algorithm for
Data-Centric Wireless Sensor Networks

本論文係提交國立臺灣大學

資訊管理學研究所作為完成碩士

學位所需條件之一部份

研究生： 王弘翕 撰

中華民國九十五年七月

謝　詞

回想起這兩年的碩士生活，是我求學過程中，付出最多心力亦是收穫最多的一段日子。而這兩年的研究生涯將隨著本篇論文的完成告一段落。

首先我要感謝我的恩師　林永松老師，學生資質駑鈍，但老師總能不厭其煩、循序漸進地啟發我的思想，一步步引領我進入學術殿堂，讓我如沐春風，一窺其中奧妙。而老師嚴謹的研究態度與治學風格，更令學生一身受用不盡。此外，承蒙孫雅麗所長、林盈達教授、呂俊賢教授與趙啟超教授在口試期間對學生的論文提出了許多建議與指正，惠我良多。

感謝博士班的演福學長在這段時間中，不斷地給我協助與指導，讓我在不知所措時能找到一個方向。也謝謝學長您在我寫論文的過程中，總能與我分享您豐富的經驗。也謝謝柏皓學長一直給我許多幫助與建議，讓我不但有機會能體驗報告時的那種壓力感，也讓我能有機會練習如何在簡短的時間內，表達出自己的想法。更感謝與恭賀與我們一齊口試的國維學長，在口試當天為我加油打氣，給我勇氣與自信。同時也謝謝去年畢業的書平、明源、孝澤、琳智學長姐在論文給予的諸多協助，以及建宏學長帶來的歡樂。也感謝研一的諸位學弟妹們：雅芳、坤道、俊維、承賓、岦毅、翊恆、怡孜，謝謝你們在口試期間的協助，讓我能專心準備口試。

此外，感謝 Eternity 的諸位戰友，在我人生最徬徨無助的那段日子，出 raid 陪我度過了無數個寂靜的夜晚，讓我在苦悶的研究生生涯中，找到了抒發心中情緒的管道。而好在 Eternity 搬了家，讓學生沒有機會再墮落下去，可以有充裕的時間來完成我的論文。

最後我要感謝我的父母：王震先生與沈嘉鳳女士，謝謝你們能默默地在背後支持、關懷並諒解我，讓我無後顧之憂。此外，我要特別感謝這段時間陪在我身邊的珮琪，謝謝她在我數度想要放棄時，總是給我鼓勵與祝福。如果沒有她的支持與鼓勵，讓我能有決心走過這一段艱苦的歷程，我想我未必能順利完成學業。

最後我要感謝過去曾經幫助我，使我成長的親人，老師，朋友與戰友們，謝謝你們。

王弘翕　　謹　識

于臺大資訊管理研究所

民國 九十五年 七月

論文摘要

論文題目：具資料集縮能力無線感測網路之低能耗與延遲排程演算法

作　　者：王弘翕

指導教授：林永松　博士

無線感測網路是由許多具有感應、計算以及無線通訊能力之感測器所組成的。由於無線感測網路通常是用隨機的方式來撒佈，故為能源即將耗盡的感測器充電是不可行的。因此，如何延長整個網路的系統壽命成為了無線感測網路相關研究中一項非常重要的議題。

本篇論文研究在感測器具有資料集縮能力之無線感測網路中，如該何路由並且排程所有感測器其活動之問題。我們針對了這個問題提出了一個數學模型。並且，更進一步地提出了一個能建立資料集縮樹以及排程無線感測網路中所有感測器其活動之演算法。藉由拉格蘭日鬆弛法，我們可以找到一個近似的最佳解並且驗證我們提出的演算法是否能達到低能耗、資料集縮能力以及保証所產生的延遲會在一個合理的範圍內。

關鍵詞：排程、資料集縮、高效率節能、低延遲、資料中心路由、最佳化、拉格蘭日鬆弛法、整數線性規劃、無線感測網路。

THESIS ABSTRACT

GRADUATE INSTITUTE OF INFORMATION MANAGEMENT

NATIONAL TAIWAN UNIVERSITY

NAME：HUNG-SHI WANG

ADVISER：YEONG-SUNG LIN

An Energy and Delay Efficient Scheduling Algorithm

for Data-Centric Wireless Sensor Networks

Wireless sensor networks (WSNs) consist of a number of small nodes with sensing, computation, and wireless communication abilities. Because of the deployment of sensors would be typically in random fashion. It would not be feasible to recharge the batteries of a moribund sensor. Hence, how to prolong the lifetime becomes a principal issue in wireless sensor networks.

In this thesis, we emphasize on a problem of routing and scheduling the activities of all sensors in a data-centric wireless sensor network. We propose a mathematical formulation to model this problem as an integer programming problem, where the objective function is to minimize the total energy consumption, including transmitting, receiving, idling and sleeping. By Lagrangean Relaxation method, we can find a near optimal solution out and verify whether the algorithm we proposed achieves energy efficiency, fulfils data aggregation, and ensures the latency within a reasonable range.

Keywords: Scheduling, Data aggregation, Energy-Efficient, Delay-Efficient, Data-centric Routing, Optimization, Lagrangean Relaxation Method, Integer Linear Programming, Wireless Sensor Network.

Table of Contents

Lists of Tables

Lists of Figures

Chapter 1 Introduction

1.1 Background

Network becomes a necessary media in our daily lives because of the profound benefit and convenience provided by Internet. The traditional network is confined by wire devices, however, it makes users or subscribers stay continually in front of their personal computers or the other communication devices to obtain the advantages of Internet. Thus, it makes the convenience of network be restricted.

During recent years, there have been more and more scholars and researchers dedicating themselves to the research in wireless communication. Due to the advance of wireless communication techniques, numerous wireless related applications have been proposed and growth rapidly. And WSNs are the primary among these applications.

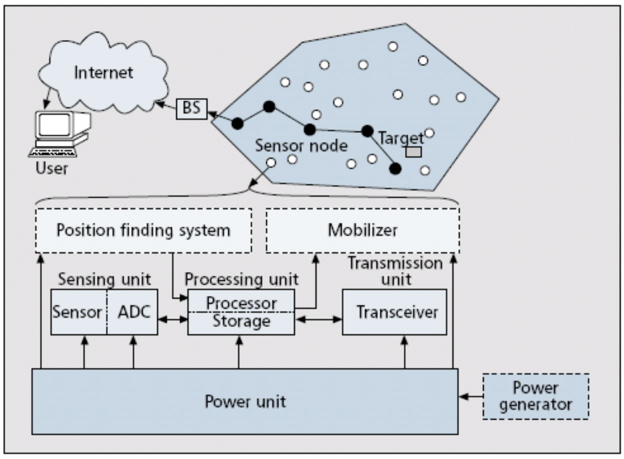

Fig.1-1 The Structure of WSNs [15]

WSNs consist of a number of small nodes with sensing, computation, and wireless communication abilities. Each sensor in the WSNs are capable of probing and collecting environmental information such as temperature, ocean current, and atmospheric pressure, and communicating with others. For example, as shown in Fig. 1-1, after the target of interest triggers an event, the sensors around the target will sense this event and forward those messages to a node with extra computation and communication power, called sink node or base station, via the predetermined routing algorithm. And sink node will transmit these messages to Internet by wired or wireless network.[15, 16]

WSNs have great effects on military and civil applications such as object tracking, disaster detection, weather monitoring and security and tactical surveillance. In all applications described above, nevertheless, the deployment of those scenarios would be typically in random fashion. Hence, recharging the batteries of a moribund sensor would not be feasible. In the past few years, therefore, to prolong the lifetime becomes a principal issue in wireless sensor networks.

1.2 Motivation

Most energy in traditional MAC protocol is wasted in idle listening. Since a sensor does not know when it will be the receiver of the message from its neighbors. Hence, it must keep its radius and hold in listen mode all the time. Some researchers propose few algorithms to schedule the activities of all sensors in order to reduce the energy consumption. Nonetheless, there is a tradeoff between energy consumption and latency.[1, 2, 4]

Furthermore, this effect will be much more significant when we take the transmission radii of sensors into consideration. For example, there are two extreme cases. The first case is that all data sources enlarge their radii as large as possible, so the sources can reach and deliver their messages to sink directly. It can minimize the delay, nonetheless, it also lead to tremendous energy expenditure in the meantime. Another case is that all sensor nodes diminish their duty-cycle as tiny as possible. Because all sensors turn off their radio and get into sleep mode, the power consumption will be minimized. However, it will result in immense sleep latency.

In Chapter 2, we propose a mathematical formulation to construct a data aggregation tree and to schedule the activities of all sensors in wireless sensor networks. The objective function is to minimize the energy consumption, and it implies that the lifetime of this network will be maximized. By Lagrangean Relaxation, we can optimally solve this problem. And it verifies whether the algorithm we proposed achieves energy efficiency, fulfils data aggregation, and ensures the latency within a reasonable range.

1.3 Literature Survey

1.3.1 Power Consumption

The issue of the power consumption is often to be discussed in WSN area, the Mac layer protocol of the initial sensors that utilized the Mac layer protocol of the 802.11 wireless communication protocols directly.

Even though this method is quite intuitive and effective, it produces idle listening that causes the power consumed hastily by the sensor nodes. It also reduces the lifetime of the entire network. Therefore, the concept of the duty-cycle that is also called S-Mac Protocol was broached [1, 2]. Ye et al. conceives of the sensor nodes are at the idle condition in the major part of the time and the model divides the sensor activities into two kinds of cycles that are the listen and the sleep phases, as shown in Fig 1-2. The sensor nodes will wait for a short time and after that will be asleep when the sensor nodes notice the information is not transmitting while at the listen circle in order to reduce the energy consumption.

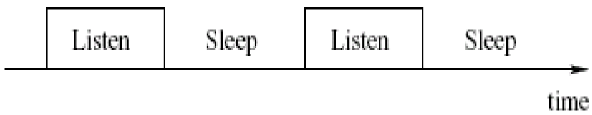

Fig.1-2 Illustration of Periodically Listen and Sleep [1]

On the other hand, Lu et al. considers the mechanism of the S-Mac is not the optimization for the complete design and advanced the T-Mac protocol theory that is denominated as timeout-Mac. Basically, the T-Mac and the S-Mac theories are using the same frame, such as exploiting the periodically sleep to avoid the idle listening, the rts/cts/data/ack to approach the collision avoidance and so on. Conversely, the most distinction of the both manners is the duty-cycle of the S-Mac is fixed but is dynamic for the T-Mac [4].

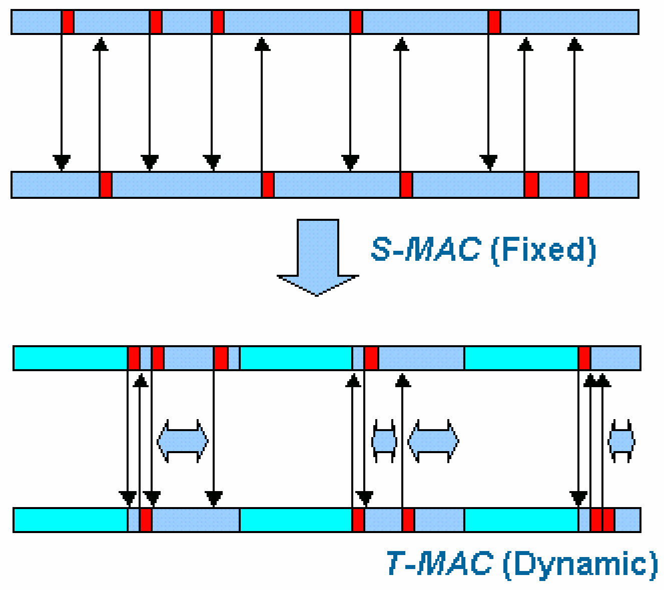

Fig 1-3. Comparison of S-MAC and T-MAC

1.3.2 Delay

According to the fore-mentioned, the S-Mac and the T-Mac can reduce energy consumption more effectively than the 802.11 Mac Protocol without question. Nevertheless, when some sensor node A wants to transmit the message to another sensor node B will find out the B has already entered the dormancy status because of the sensor nodes shut down the communicable medium unceasingly to facilitate the diminution of the energy consumption effectively. Therefore, the sensor node A has to wait until the sensor node B wake up in order to process the transmission of the information, that the gap of the period is called sleep latency [1, 2].

In the research of Ye et al. indicates the percentage of the energy consumption and the sleep latency will be an index function with the evolve direction in the WSN area. For the purpose of decreasing more energy consumption, we must endure longer sleep latency. There are a various subtle tradeoff between energy saving and end-to-delay.

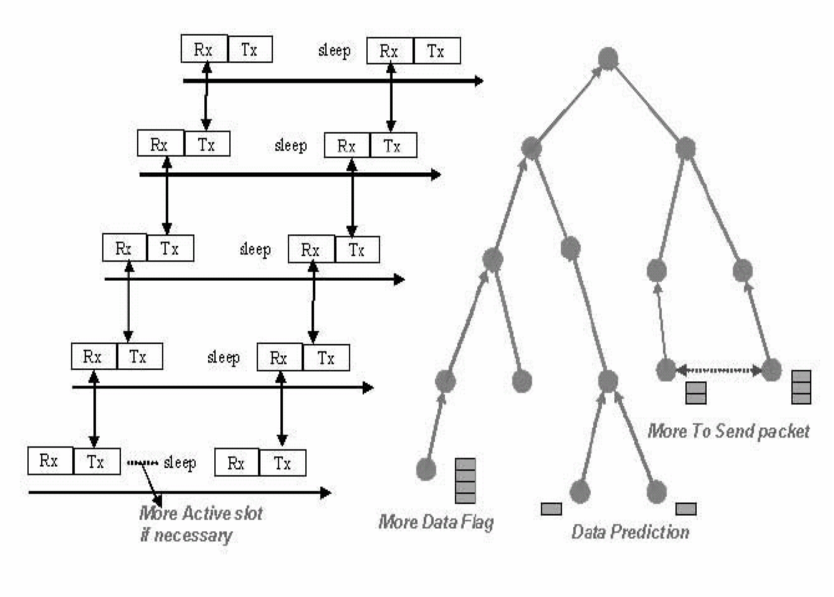

Fig 1-4. The Data Gathering Tree Structure of D-MAC [3]

Lu et al. of the D-Mac scheme agrees with the point of view, thought the D-Mac scheme identify that certain systems will require the minimum power consumption, others organization will possibly necessitate the minimum end-to-end delay. Thereupon, they regard the duty-cycle as a given requirement that achieves the minimum end-to-end delay under the established power consumption.[3, 5]

As shown in Fig 1-4, in order to propose a energy and delay efficient Mac protocol, Lu et al. developed a special structure, namely data gathering tree. Without doubt, D-MAC could achieve energy-efficient and low latency. However, it can't adopt data aggregation simultaneously

1.3.3 Data Aggregation and Data-centric Routing

In order to reduce the transmission cost, several scholars propose a concept, namely data aggregation. The idea of data aggregation is to combine the data coming from different sources as possible as its can. By eliminating the redundancy, minimizing the number of transmissions, data aggregation could fulfill the goal of energy saving. Therefore, data aggregation has put forward as a particularly useful paradigm for wireless routing in Sensor networks.[11, 12, 14, 16]

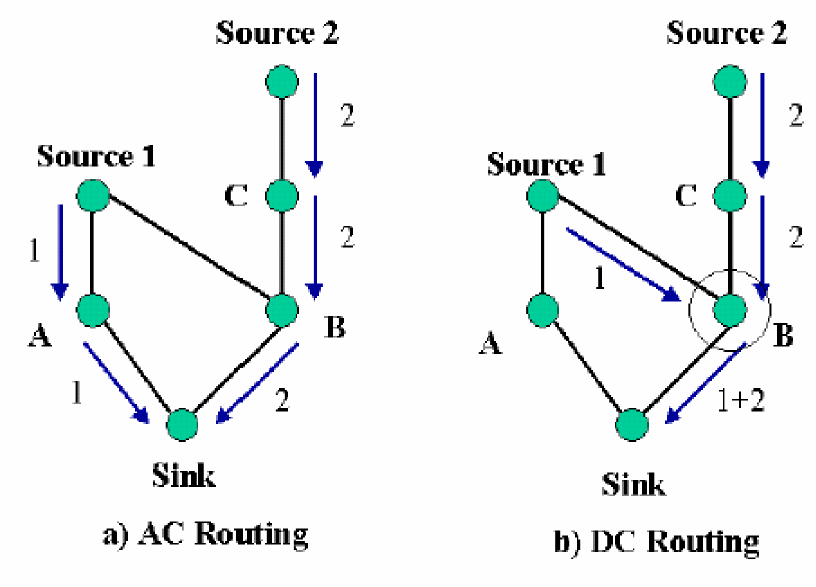

Fig 1-5. Illustration of AC versus DC Routing [13]

There are two different manners when data is send from every source to the sink:

Address-centric Routing (AC Routing):

Each source sends its information along the shortest path to the sink node.

Data-centric Routing (DC Routing):

Each source sends its information along the path where either a shortest path or a path used by its neighbor.

As shown in Fig 1-5, there are two sources in the network. In the AC routing scheme, Source1 will deliver its data to sink by node A. In the DC routing scheme, Source will transfer its data to node B rather than A. Node B will aggregate the flows coming from A and B, then relay the condensed data to the sink node. Krishnamachari et al propose several sub-optimal routing algorithms, namely Shortest Path Tree (SPT), Center Nearest Source (CNS), and Greedy Incremental Tree (GIT), for this data-centric routing problem. [13, 14]

Generally speaking, for the most of the existing routing algorithms, they didn't take idle listening and scheduling problem into consideration. For the of existent Mac protocols, they cant achieve energy efficient, delay efficient, data aggregation in chorus. Thus, that is the reason why we are dedicating ourselves to this research.

Chapter 2 Problem Formulation

2.1 Problem Description

As shown in Fig 2-1 and Fig 2-2, the problem to be solved is to decide how to construct a data aggregation tree and how to schedule the activities of all nodes that are involved in the data aggregation tree to minimize the total energy consumption. It can be applied to periodic application scenario where each sensor node periodically senses and reports information to the sink node such as weather monitoring.

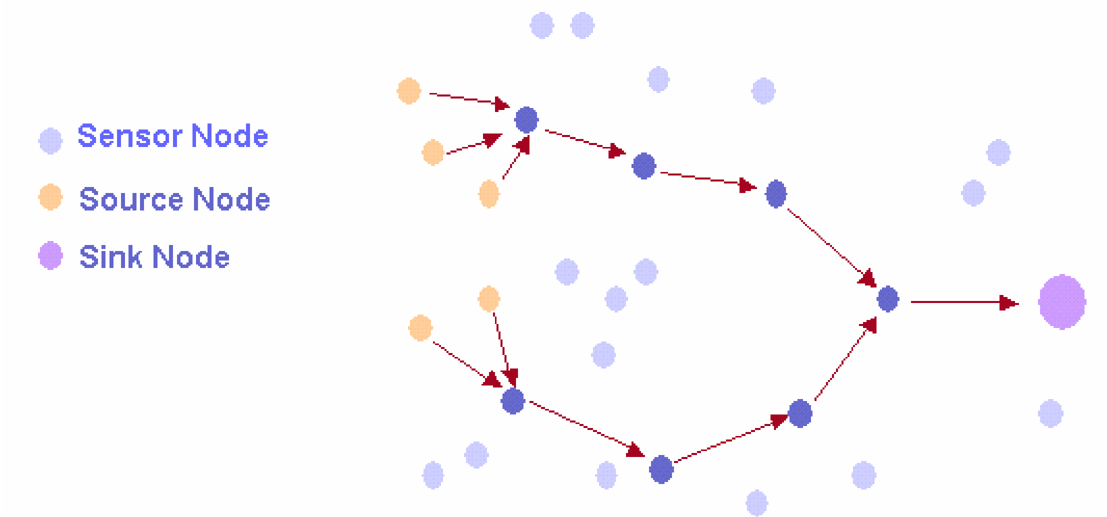

Fig 2-1. Illustration of The Data Aggregation Tree Construction

In this model, we assume there is a centralized node to determine the activities of all sensors and this network is synchronized strictly. By this assumption, we can reasonably presume the propagation delay could be ignored. Hence, the link delay could be considered as a constant.

As shown in Fig 2-2,we divide the time into several slots. To achieve collision free, moreover, there should be at most one node be allowed to communicate with others in the same interfere area. Each sensor will be assigned a slot that permits to relay the sensed data to his neighbor. By adjusting *n*, *m*, and *w*, the receiving sending, and sleeping behavior of all nodes could be organized orderly.

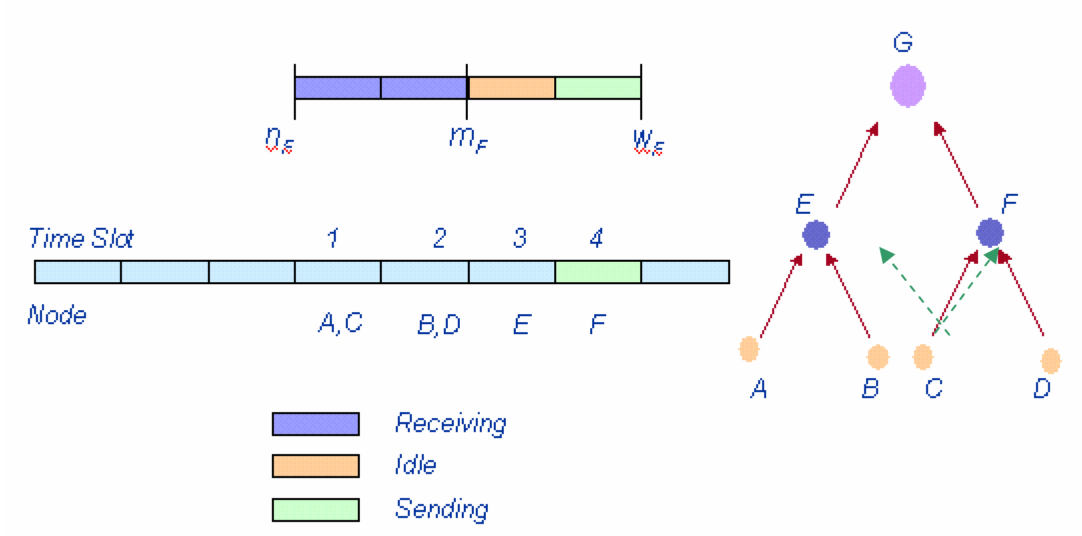

Fig 2-2. Slots Assignment and The Activities of Sensors.

However, to schedule the activities of entire networks is not a easy task. Because of the transmission radius of each sensor is dynamic, we cant determine the connectivity of entire network beforehand. And whether a node will disturb its neighbors is also a variable and can’t be predictable. And it is the major difficulty of this problem. The summary of problem description is listed as below.

Table 2.1 Problem Description

Assumptions*:** 1. There is a centralized node to determine the activities of all sensor nodes. 2. A network with strict synchronization 3. Propagation delay can be ignored. 4. Link delay can be considered as a constant. ***Given*:** 1. The set of all sensor nodes 2. The set of all candidate paths for each data source to reach sink node 3. The set of all data sources 4. Longest hops along shortest path from sink node to reach the farthest data source 5. An arbitrary large number M 6. The sink node 7. Maximum End-to-end delay requirement T ***Objective: To minimize the energy consumption of the entire wireless sensor networks.

Subject to:

1. Routing constraint– each data source should only select one routing path to send data to the sink node.
2. Tree constraint – the combination of routing path of each data source shall be a tree, namely data aggregation tree.
3. Scheduling constraint – the sleeping, idleness, receiving and sending behavior of all nodes should be considered.
4. Number of Neighbors constraint – the total number of neighbors whose transmission radius and timing covers the other nodes should be considered.
5. Collision Free constraint –the number of communication nodes among a node is at most one.

To determine:

1. Routing path for each data source
2. Transmission radius for each sensor node
3. Whether a link should be on the data aggregation tree
4. The data aggregation tree
5. The wake up time of each sensor node on data aggregation tree
6. The aggregation complete time of each sensor node on data aggregation tree
7. The transmission finish time of each sensor node on data aggregation tree

2.2 Problem Notation

Table 2.2 Notation descriptions for give parameters

Given Parameters	
Notation	Description
V	The set of sensor nodes
P_s	The set of all possible paths from the data source s node to the sink node.
S	The set of all data source nodes
H	Longest distance of shortest path to reach farthest data source node
M	An arbitrary large number
q	The sink node
T	End-to-end delay requirement
R	The set of all possible transmission radii that sensor node can adopt
$e(r_u)$	Energy consumption function of node u, which is a function of sensor's transmission radius
V	Data volume of a message, which is a constant
K	Processing cost of each incoming message, which is a constant
E_r	Energy consumption rate when sensor nodes are receiving
E_{idle}	Energy consumption rate when sensor nodes are idle
E_{sleep}	Energy consumption rate when sensor nodes are sleeping
$\delta_{p(uv)}$	The indicator function which is 1 if the link (u, v) is on the path p and 0 otherwise

Table 2.3 Notation descriptions for decision variables

Decision Variables	
Notation	Description
x_p	1 if the data source node uses the path *p* to reach the sink node
y_{uv}	1 if the link (*u*, *v*) is on the tree
r_u	The transmission radius of node *u*
n_v	When the node *v* must wake up
m_v	When the node *v* will complete its aggregation
w_v	When the node *v* could finish its transmission and turn off hits raido
ϕ_{uv}	1 if the node *v* is covered within transmission radius of the node *u*
z_{uv}	1 if the node *v* will be interfered by the node *u*
z_{uv1}	1 if the maximum end-to-end delay from leaf nodes to node *u* is large than the minimum begin time of all flows to node *v*.
z_{uv2}	1 if the maximum end-to-end delay from leaf nodes to node *v* is large than the minimum begin time of all flows to node *u*.
d_{uv}	The difference between w_u and w_v

2.3 Problem Formulation

Objective Function

$$Z_{IP} = \min \sum_{u \in V} \{(m_u - n_u)E_r + (w_u - 1 - m_u)E_{idle} + [T - (m_u - n_u)]E_{sleep} + K \sum_{v \in V} y_{vu} + V \cdot e(r_u)\}$$

(IP)

subject to:

Routing Constraints

$$\sum_{p \in P_s} x_p = 1 \qquad \forall s \in S \qquad (2.1)$$

$$\sum_{p \in P_s} x_p \delta_{p(uv)} \leq y_{(uv)} \qquad \forall s \in S \;\; u,v \in V \qquad (2.2)$$

$$x_p = 0 \;\; or \;\; 1 \qquad \forall p \in P_s \;\; s \in S \qquad (2.3)$$

Tree Constraints

$$\sum_{u \in V} \sum_{v \in V} y_{(uv)} \geq \max\left\{ H, \;\; |S| \right\} \qquad (2.4)$$

$$\sum_{s \in S} \sum_{p \in P_s} x_p \delta_{p(uv)} \leq |S| y_{(uv)} \qquad \forall u,v \in V \qquad (2.5)$$

$$\sum_{v \in V} y_{(uv)} \leq 1 \qquad \forall u \in V \qquad (2.6)$$

$$\sum_{u \in V} y_{(su)} = 1 \qquad \forall s \in S \qquad (2.7)$$

$$\sum_{u \in V} y_{(uq)} \geq 1 \qquad (2.8)$$

$$y_{(uv)} = 0 \;\; or \;\; 1 \qquad \forall u,v \in V \qquad (2.9)$$

Scheduling Constraints

$$w_u - T(1 - y_{(uv)}) \leq m_v \qquad \forall u,v \in V \qquad (2.10)$$

$$n_v \leq m_u + T(1 - y_{(uv)}) \qquad \forall u,v \in V \qquad (2.11)$$

$$m_u + 1 \le w_u \qquad \forall u \in V \qquad (2.12)$$

$$m_u \in \{0,1,2,\ldots,T\} \qquad \forall u \in V \qquad (2.13)$$

$$n_u \in \{0,1,2,\ldots,T\} \qquad \forall u \in V \qquad (2.14)$$

$$w_u \in \{0,1,2,\ldots,T\} \qquad \forall u \in V \qquad (2.15)$$

Number of Neighbors Constraints

$$\frac{r_u - \eta_{uv}}{M} \le \phi_{uv} \qquad \forall u, v \in V \qquad (2.16)$$

$$\phi_{uv}\eta_{uv} \le r_u \qquad \forall u, v \in V \qquad (2.17)$$

$$y_{(uv)} \le \phi_{uv} \qquad \forall u, v \in V \qquad (2.18)$$

$$r_u \in R \qquad \forall u \in V \qquad (2.19)$$

$$r_s \ne 0 \qquad \forall s \in S \qquad (2.20)$$

$$\phi_{(uv)} = \ 0 \ \ or \ \ 1 \qquad \forall u, v \in V \qquad (2.21)$$

$$\frac{m_v - w_u}{M} \le z_{uv1} \qquad \forall u, v \in V \qquad (2.22)$$

$$\frac{w_u - m_v}{M} \le 1 - z_{uv1} \qquad \forall u, v \in V \qquad (2.23)$$

$$\frac{w_u - n_v}{M} \le z_{uv2} \qquad \forall u, v \in V \qquad (2.24)$$

$$\frac{n_v - w_u}{M} \le 1 - z_{uv2} \qquad \forall u, v \in V \qquad (2.25)$$

$$w_u - w_v \le D_{(uv)} \qquad \forall u, v \in V \qquad (2.26)$$

$$w_v - w_u \le D_{(uv)} \qquad \forall u, v \in V \qquad (2.27)$$

$$1 - z_{uv} \le D_{(uv)} \qquad \forall u, v \in V \qquad (2.28)$$

$$(z_{uv1} + z_{uv2} + \phi_{uv}) - 2 \le z_{(uv)} \qquad \forall u, v \in V \qquad (2.29)$$

$$z_{uv1} = \ 0 \ \ or \ \ 1 \qquad \forall u, v \in V \qquad (2.30)$$

$$z_{uv2} = 0 \ or \ 1 \qquad \forall u,v \in V \qquad (2.31)$$

$$D_{(uv)} \in \{0,1,2,...,T\} \qquad \forall u,v \in V. \qquad (2.32)$$

Collision free Constraints

$$\sum_{u \in V} z_{(uv)} \le 1 \qquad \forall v \in V \qquad (2.33)$$

$$z_{(uv)} = 0 \ or \ 1 \qquad \forall u,v \in V \qquad (2.34)$$

Explanation of the objective function:

The objective function of (IP) is to minimize total energy consumption that includes receiving, idle, sleeping and sending. And it also implies that the lifetime of this network will be maximized.

Explanation of constraints

[1] Routing Constraints

Constraints (2.1) and (2.3) confine that for each data source, it should be assigned exactly one routing path to ensure the sensed data will be transmitted to the sink node.

Constraint (2.2) confines that if the path *p* is selected, all links involved in *p* must be on the aggregation tree.

[2] Tree Constraints

Constraints (2.4) and (2.9) confine that that total number of links on the aggregation tree is at least the maximum of H and the cardinality of S.

Constraint (2.5) confines that the union of the routing paths destined for the sink node does exist a cycle.

Constraint (2.6) confines that all intermediate nodes on data aggregation tree should have at most one outgoing link.

Constraint (2.7) confines that every data source should have exactly one outgoing link.

Constraint (2.8) confines that all flows coming from leaf nodes will eventually converge on sink node.

[3] Scheduling Constraints

m_v is when all incoming flows from leaf nodes to node *v* will be aggregated entirely. Constraint (10) confines that a node must stay in receive mode until all of its children terminate their transmission. Therefore, when a node will finished its aggregation is at least the time all its children complete their transmission.

Constraint (11) confines that if link (*u*,*v*) is involved in aggregation tree, then the begin time of node *u* shall less than the time aggregation complete time of its descendants. In another word, if there is a message from node *u* to node *v*, then node *v* should wake up before node *u* finish its aggregation. Otherwise, it will lose this message.

Constraint (12) confines that a node will never enter the dormancy status before it receive all messages coming from its offspring.

[4] Number of Neighbors Constraints

Constraints (2.16)~(2.29) are number of neighbors constraints. $\sum_{u \in V} z_{(uv)}$ stands for the total number of sensor nodes interfered by the transmission of sensor node u, or the total number of sensor nodes whose transmission interfere sensor node v.

If node u is covered within the transmission radius of node v ($r_u \geq \eta_{uv}$), and there is any overlap between the communication of node u and node v ($n_v \leq w_u \leq m_v$ or $w_v = w_u$), z_{uv} should be 1. By introducing z_{uv1} and z_{uv2}, we can model the relationship between ϕ_{uv}, n_v, m_v, w_u, and z_{uv} properly.

ϕ_{uv} stands for whether the node v is covered within transmission radius of the node u. If $r_u \geq \eta_{uv}$, ϕ_{uv} should be equal to 1 and 0 otherwise. By jointly enforcing constraint (2.16) and (2.17), we can model the relationship between r_u, η_{uv}, and ϕ_{uv}. .These two constraints are complementary as shown in Table 2.4.

Table 2.4 Explanation of Constraints (2.16) and (2.17)

ϕ_{uv}	Constrain (2.16)	Constrain (2.17)
$r_u \geq \eta_{uv}$	ϕ_{uv} =1	ϕ_{uv} =0,1
$r_u < \eta_{uv}$	ϕ_{uv} =0,1	ϕ_{uv} =0

Constraint (2.18) confines that if $y_{(uv)}$ equals to 1 then z_{uv} also must be 1.

z_{uv1} stands for whether m_v is larger than w_u or not. If $m_v \geq w_v$, z_{uv1} should be equal to 1 and 0 otherwise. By jointly enforcing constraint (2.22) and (2.23), we can well model the relationship described above

By the same token, z_{uv2} stands for whether w_u is larger than n_v or not or not. By jointly enforcing constraint (2.24) and (2.25), we can model the relationship between w_u , n_v ,and z_{uv2} . These four constraints are complementary as shown in Table 2.5

Table 2.5 Explanation of Constraints (2.11)~(2.14)

z_{uv1}	Constrain (2.22)	Constrain (2.23)
$m_v \geq w_u$	$z_{uv1} = 1$	$z_{uv1} = 0,1$
$m_v < w_u$	$z_{uv1} = 0,1$	$z_{uv1} = 0$
z_{uv2}	Constrain (2.24)	Constrain (2.25)
$w_u \geq n_v$	$z_{uv2} = 1$	$z_{uv2} = 0,1$
$w_u < n_v$	$z_{uv2} = 0,1$	$z_{uv2} = 0$

Constraint (2.29) confines that if node *u* is covered within the transmission radius of node *v*, and there is any overlap between the communication of node *u* and node *v*, z_{uv} shall be one. In another word, If $\phi_{uv} = 1$, $z_{uv1} = 1$, and $z_{uv2} = 1$, z_{uv} should be equal to 1 and 0 otherwise. As shown in Table 2.6.

Table 2.6 Explanation of Constraint (2.29)

z_{uv}	Constrain (2.29)
$z_{uv1} + z_{uv2} + \phi_{uv} = 0$	$z_{uv} = 0,1$
$z_{uv1} + z_{uv2} + \phi_{uv} = 1$	$z_{uv} = 0,1$
$z_{uv1} + z_{uv2} + \phi_{uv} = 2$	$z_{uv} = 0,1$
$z_{uv1} + z_{uv2} + \phi_{uv} = 3$	$z_{uv} = 1$

Up to present, we have formulated if $n_v \leq w_u \leq m_v$ and $\phi_{uv} = 1$ then z_{uv} shall be 1. However, if $w_u = w_v$ and $\phi_{uv} = 1$ then $z_{uv1} = 0$, $z_{uv2} = 1$, and constraint (2.29) cant enforce z_{uv} to be 1(z_{uv} can be 0 or 1). Hence we need some extra constraints to restrict that if $w_u = w_v$ and $\phi_{uv} = 1$, z_{uv} will be equal to one.

By jointly enforcing constraints (2.26), (2.27) (2.28) and (2.34), we can suitably model the relationship described above.

[5] Collision free Constraints

In wireless communication environment, each sensors nodes will potentially interfere with his neighbors. Constraint (2.33) confines that the number of communication nodes among node *v* is at most one. It implies that there should be no any collision while node *v* is communicating with the sensors in the neighborhood of himself.

[6] Boundary Constraints:

Constraints (2.3), (2.9), (2.13), (2.14), (2.15), (2.21), (2.30), (2.31) and (2.32) are the integer constraints of decision variables.

Constraint (2.20) confines that the transmission radius of all data sources can not be 0.

Chapter 3 Solution Approach

3.1 Introduction to the Lagrangean Relaxation Method

Lagrangean relaxation method was widely used for scheduling and solving integer programming problems in the 1970s, because it is flexible and provides excellent solutions for these problems. Hence, it has become one of the best tools for solving optimization problems, such as integer programming, linear programming with combinatorial objective function, and non-linear programming. [7, 8, 9, 10]

By adopting Lagrangean relaxation method, there are several advantages. In the beginning, we relax the complicated constraints of the primal mathematical formulation and form a new Lagrangean relaxation problem in many different ways. By relaxing the complicated constraints and putting them in the objective function with the corresponding Lagrangean multipliers, we can divide the original problem into several independent and easily solvable sub-problems. After that, for each sub-problem, we detect the underlying structure and properties and solve it optimally in some well-known algorithms.

By solving the Lagragaen relaxation problem, we can get a boundary to the objective function of the original problem. The solution of the Lagrangean relaxation problem is always a lower bound (to the original minimization problem). Then, we use the boundary to design a heuristic approach to get a primal feasible solution. To solve the original problem optimally and minimize the gap between the primal problem and the Lagrangean relaxation problem, we improve the lower bound by solving the sub-problems optimally and using the subgradient method to adjust the multipliers at each iteration.

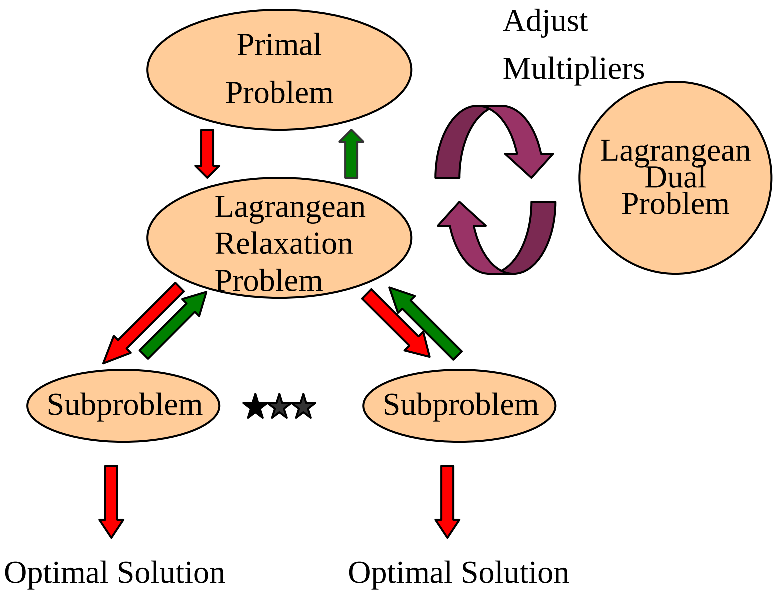

Figure 3.1 Illustration of the Lagrangean Relaxation Method

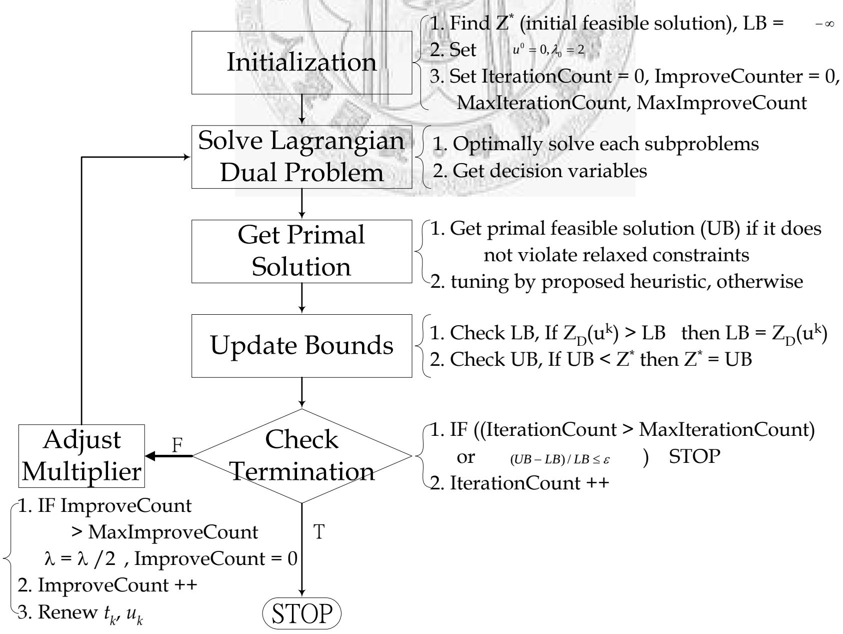

Figure 3.2 Procedures of the Lagrangean Relaxation Method

3.2 Lagrangean Relaxation

In (IP), by introducing Lagrangean multiplier vectors u^1, … , u^{16}, we dualize Constraints (2.2), (2.5), (2.10), (2.11), (2.12), (2.16), (2.17), (2.18), (2.22), (2.23), (2.24) ,(2.25), (2.26), (2.27) (2.28) and (2.29) to obtain the following Lagrangean relaxation problem (LR).

$$Z_{LR}(u^1, u^2, u^3, u^4, u^5, u^6, u^7, u^8, u^9, u^{10}, u^{11}, u^{12}, u^{13}, u^{14}, u^{15}, u^{16},) =$$

$$\min \sum_{u\in V}[(m_u - n_u)E_r + (w_u - 1 - m_u)E_{idle} + [T - (w_u - n_u)]E_{sleep} + K\sum_{v\in V} y_{(vu)} + V \cdot e(r_u)] +$$

$$\sum_{s\in S}\sum_{u\in V}\sum_{v\in V}\mu^1_{suv}\left[\sum_{p\in P_s} x_p \delta_{p(uv)} - y_{(uv)}\right] +$$

$$\sum_{u\in V}\sum_{v\in V}\mu^2_{uv}\left[\sum_{s\in S}\sum_{p\in P_s} x_p \delta_{p(uv)} - |S| y_{(uv)}\right] + q$$

$$\sum_{u\in V}\sum_{v\in V}\mu^3_{uv}\left[w_u - T(1 - y_{(uv)}) - m_v\right] +$$

$$\sum_{u\in V}\sum_{v\in V}\mu^4_{uv}\left[n_v - T(1 - y_{(uv)}) - m_u\right] + +$$

$$\sum_{u\in V}\sum_{v\in V}\mu^5_{uv}\left[m_u + 1 - w_u\right] + +$$

$$\sum_{u\in V}\sum_{v\in V}\mu^6_{uv}\left[(r_u - \eta_{uv}) - M\phi_{uv}\right] + +$$

$$\sum_{u\in V}\sum_{v\in V}\mu^7_{uv}\left[(\eta_{uv}\phi_{uv} - r_u)\right] + +$$

$$\sum_{u\in V}\sum_{v\in V}\mu^8_{uv}\left[(y_{uv} - \phi_{uv})\right] + +$$

$$\sum_{u\in V}\sum_{v\in V}\mu^9_{uv}\left[(m_v - w_u) - Tz_{uv1}\right] + +$$

$$\sum_{u\in V}\sum_{v\in V}\mu^{10}_{uv}\left[(w_u - m_v) - T + Tz_{uv1}\right] + +$$

$$\sum_{u\in V}\sum_{v\in V}\mu^{11}_{uv}\left[(w_u - n_v) - Tz_{uv2}\right] + +$$

$$\sum_{u\in V}\sum_{v\in V}\mu^{12}_{uv}\left[(n_v - w_u) - T + Tz_{uv2}\right] + +$$

$$\sum_{u\in V}\sum_{v\in V}\mu^{13}_{uv}\left[(w_u - w_v) - D_{uv}\right] + +$$

$$\sum_{u\in V}\sum_{v\in V}\mu_{uv}^{14}\left[(w_v - w_u) - D_{uv}\right] + +$$

$$\sum_{u\in V}\sum_{v\in V}\mu_{uv}^{15}\left[1 - z_{uv} - D_{uv}\right] + +$$

$$\sum_{u\in V}\sum_{v\in V}\mu_{uv}^{16}\left[(z_{uv1} + z_{uv2} + \phi_{uv}) - 2 - z_{uv}\right] \tag{LR}$$

Subject to:

$$\sum_{p\in P_s} x_p = 1 \qquad \forall s \in S \tag{3.1}$$

$$\sum_{u\in V}\sum_{v\in V} y_{(uv)} \geq \max\left\{ H, \ |S| \right\} \tag{3.2}$$

$$\sum_{v\in V} y_{(uv)} \leq 1 \qquad \forall v \in V \tag{3.3}$$

$$\sum_{u\in V} y_{(su)} = 1 \qquad \forall s \in S \tag{3.4}$$

$$\sum_{u\in V} y_{(uq)} \geq 1 \tag{3.5}$$

$$\sum_{u\in V} z_{uv} \leq 1 \qquad \forall v \in V \tag{3.6}$$

$$x_p = 0 \ \ or \ \ 1 \qquad \forall p \in P_s \ \ s \in S \tag{3.7}$$

$$y_{(uv)} = 0 \ \ or \ 1 \qquad \forall u, v \in V \tag{3.8}$$

$$m_u \in \{0,1,2,\ldots,T\} \qquad \forall u \in V \tag{3.9}$$

$$n_u \in \{0,1,2,\ldots,T\} \qquad \forall u \in V \tag{3.10}$$

$$w_u \in \{0,1,2,\ldots,T\} \qquad \forall u \in V \tag{3.11}$$

$$z_{uv1} = \ 0 \ \ or \ \ 1 \qquad \forall u, v \in V \tag{3.12}$$

$$z_{uv2} = \ 0 \ \ or \ \ 1 \qquad \forall u, v \in V \tag{3.13}$$

$$z_{(uv)} = \ 0 \ \ or \ \ 1 \qquad \forall u, v \in V \tag{3.14}$$

$$r_u \in R \qquad \forall u \in V \tag{3.15}$$

$$r_s \neq 0 \qquad \forall s \in S \tag{3.16}$$

$$\phi_{(uv)} = 0 \ \ or \ \ 1 \qquad \forall u, v \in V \tag{3.17}$$

$$D_{uv} \in \{0,1,2,...,T\} \qquad \forall u, v \in V \text{ .} \tag{3.18}$$

We can decompose (LR) into eleven independent subproblems.

3.2.1 Subproblem 1 (related to decision variable x_p)

$$\min \sum_{s\in S}\sum_{u\in V}\sum_{v\in V}\sum_{p\in P_s}(\mu_{suv}^1+\mu_{uv}^2)x_p\delta_{p(uv)} \qquad \text{(SUB 3.1)}$$

subject to:

$$\sum_{p\in P_s} x_p = 1 \qquad \forall s\in S \qquad (3.1)$$

$$x_p = 0 \ \ or \ \ 1 \qquad \forall p\in P_s \ \ s\in S \ . \qquad (3.7)$$

(SUB 3.1) can be further decomposed into $|S|$ independent shortest path problems with nonnegative arc weights. For each shortest path problem, it can be easily solve by Dijkstra's algorithm. The computational complexity of Dijkstra's algorithm is $O\left(|N|^2\right)$ for each destination of source node.

3.2.2 Subproblem 2 (related to decision variable y_{uv})

$$\min \sum_{u\in V}\sum_{v\in V}\left\{\left[\mu_{uv}^8+(\mu_{uv}^3+\mu_{uv}^4)T-\sum_{s\in S}\mu_{suv}^1-\mu_{uv}^2\,|S|+K\right]y_{(uv)}\right\} \qquad \text{(SUB 3.2)}$$

Subject to:

$$\sum_{u\in V}\sum_{v\in V} y_{uv} \geq \max\left\{\, H,\ \ |S| \,\right\} \qquad (3.2)$$

$$\sum_{v\in V} y_{uv} \leq 1 \qquad \forall u\in V \qquad (3.3)$$

$$\sum_{u\in V} y_{(su)} = 1 \qquad \forall s\in S \qquad (3.4)$$

$$\sum_{u\in V} y_{(uq)} \geq 1 \qquad (3.5)$$

$$y_{(uv)} = 0 \ \ or \ 1 \qquad \forall u,v\in V \ . \qquad (3.8)$$

The proposed algorithm for solving (SUB 3.2) is described as follows:

Step 1: For each link (u,v) compute the coefficient $\mu_{uv}^{8}+(\mu_{uv}^{3}+\mu_{uv}^{4})T-\sum_{s\in S}\mu_{suv}^{1}-\mu_{uv}^{2}\left|S\right|+K$ for each y_{uv}.

Step 2: For all outgoing links of node u, find the smallest coefficient. If the smallest coefficient is negative then set the corresponding y_{uv} to be 1 and the other outgoing links y_{uv} to be 0, otherwise set all outgoing link y_{uv} to be 0. Repeat step 2 for all nodes.

Step 3:For all data source node s, check whether there is a outgoing link from s to the other node. If there isn't any outgoing link, find the link with smallest coefficient then set this y_{su} to be 1. Repeat step 3 for all sources.

Step 4: If the total number of y_{uv} whose value is 1 (denotes as N) are smaller than *max* {h, |S|}, then identify the nodes that have all its outgoing links y_{uv} = 0. From these identified nodes, selected (*max* {h, |S|}-N) number of these identified nodes whose corresponding coefficient are the smallest. Then, assign the outgoing link =1 with the smallest coefficient for each of these selected nodes.

The computational complexity of this algorithm is $O\left(|N|^{2}\right)$.

3.2.3 Subproblem 3 (related to decision variable m_v)

$$\min \sum_{u\in V}\left\{m_u\cdot E_r-m_u\cdot E_{idle}-m_u\cdot E_{sleep}+\sum_{v\in V}\left[\left(\mu_{uv}^{9}-\mu_{uv}^{10}-\mu_{uv}^{3}\right)m_v-\mu_{uv}^{4}m_u+\mu_{uv}^{5}m_u\right]\right\} \quad \text{(SUB 3.3)}$$

Subject to:

$$m_v\in\{0,1,2,\ldots,T\} \qquad \forall v\in V. \qquad (3.9)$$

By transforming, we can rewrite the objective function of (SUB 3.3) into another form in order that this subproblem can be efficiently solved.

Transformation:

$$\sum_{u\in V}\left\{m_u\cdot E_r - m_u\cdot E_{idle} - m_u\cdot E_{sleep} + \sum_{v\in V}\left[\left(\mu_{uv}^9 - \mu_{uv}^{10} - \mu_{uv}^3\right)m_v - \mu_{uv}^4 m_u + \mu_{uv}^5 m_u\right]\right\}$$

$$= \sum_{u\in V}(E_r - E_{idle} - E_{sleep})m_u - \sum_{u\in V}\sum_{v\in V}\left(\mu_{uv}^4 + \mu_{uv}^5\right)m_u + \sum_{u\in V}\sum_{v\in V}\left(\mu_{uv}^9 - \mu_{uv}^{10} - \mu_{uv}^3\right)m_v$$

$$= \sum_{u\in V}(E_r - E_{idle} - E_{sleep})m_u - \sum_{u\in V}\sum_{v\in V}\left(\mu_{uv}^4 + \mu_{uv}^5\right)m_u + \sum_{u\in V}\sum_{v\in V}\left(\mu_{vu}^9 - \mu_{vu}^{10} - \mu_{vu}^3\right)m_u$$

$$= \sum_{u\in V}\left\{E_r - E_{idle} - E_{sleep} + \sum_{v\in V}\left(\mu_{vu}^9 - \mu_{vu}^{10} - \mu_{vu}^3 - \mu_{uv}^4 + \mu_{uv}^5\right)\right\}m_u$$

After transforming, we can decompose (SUB 3.3) into $|V|$ independent subproblems. For each node *u*,

$$\min\left\{E_r - E_{idle} - E_{sleep} + \sum_{v\in V}\left(\mu_{vu}^9 - \mu_{vu}^{10} - \mu_{vu}^3 - \mu_{uv}^4 + \mu_{uv}^5\right)\right\}m_v \qquad \text{(SUB 3.3.1)}$$

Subject to:

$$m_u \in \{0,1,2,\ldots,T\}$$

For each (SUB 3.3.1), we calculate the coefficient $E_r - E_{idle} - E_{sleep} + \sum_{v\in V}\left(\mu_{vu}^9 - \mu_{vu}^{10} - \mu_{vu}^3 - \mu_{uv}^4 + \mu_{uv}^5\right)$ of each node *u*. If the coefficient of node *u* is negative, then set m_u to be *T*, otherwise 0. The computational complexity of (SUB 3.3.1) is $O(1)$ for each node *u*.

3.2.4 Subproblem 4 (related to decision variable n_v)

$$\min \sum_{u \in V} \left\{ -n_u E_r + n_u E_{sleep} + \sum_{v \in V} (-\mu_{uv}^{11} + \mu_{uv}^{12} + \mu_{uv}^{4}) n_v \right\} \quad \text{(SUB 3.4)}$$

Subject to:

$$n_v \in \{0,1,2,...,T\} \qquad \forall v \in V . \quad (3.10)$$

By transforming, we can rewrite the objective function of (SUB 3.4) into another form in order that this subproblem can be efficiently solved.

Transformation:

$$\sum_{u \in V} \left\{ -n_u E_r + n_u E_{sleep} + \sum_{v \in V} (-\mu_{uv}^{11} + \mu_{uv}^{12} + \mu_{uv}^{4}) n_v \right\}$$

$$= \sum_{u \in V} \left\{ -E_r + E_{sleep} + \sum_{v \in V} (-\mu_{vu}^{11} + \mu_{vu}^{12} + \mu_{vu}^{4}) \right\} n_u$$

Therefore, (SUB 3.4) can be further decomposed into $|V|$ independent subproblems. For each node *v*,

$$\min \left\{ -E_r + E_{sleep} + \sum_{v \in V} (-\mu_{vu}^{11} + \mu_{vu}^{12} + \mu_{vu}^{4}) \right\} n_u \quad \text{(SUB 3.4.1)}$$

Subject to:

$$n_u \in \{0,1,2,...,T\}$$

For each (SUB 3.4.1), we calculate the coefficient $-E_r + E_{sleep} + \sum_{v \in V} (-\mu_{vu}^{11} + \mu_{vu}^{12} + \mu_{vu}^{4})$ of each node *u*. If the coefficient of node *u* is negative, then set n_u to be *T*, otherwise 0. The computational complexity of (SUB 3.4.1) is $O(1)$ for each node *u*.

3.2.5 Subproblem 5 (related to decision variable w_v)

$$\min \sum_{u\in V}\left\{(E_{idle}-E_{sleep})w_u+\sum_{v\in V}\left[(-\mu_{uv}^{13}+\mu_{uv}^{14})w_v+(-\mu_{uv}^{9}+\mu_{uv}^{10}+\mu_{uv}^{11}-\mu_{uv}^{12}+\mu_{uv}^{13}-\mu_{uv}^{14}+\mu_{uv}^{3}-\mu_{uv}^{5})w_u\right]\right\} \tag{SUB 3.5}$$

Subject to:

$$w_v \in \{0,1,2,...,T\} \qquad \forall v \in V. \tag{3.11}$$

By transforming, we can rewrite the objective function of (SUB 3.5) into another form in order that this subproblem can be efficiently solved.

Transformation:

$$\sum_{u\in V}\left\{(E_{idle}+E_{sleep})w_u+\sum_{v\in V}\left[(-\mu_{uv}^{13}+\mu_{uv}^{14})w_v+(-\mu_{uv}^{9}+\mu_{uv}^{10}+\mu_{uv}^{11}-\mu_{uv}^{12}+\mu_{uv}^{13}-\mu_{uv}^{14}+\mu_{uv}^{3}-\mu_{uv}^{5})w_u\right]\right\}$$

$$=\sum_{u\in V}\left\{(E_{idle}+E_{sleep})w_u+\sum_{v\in V}\left[(-\mu_{vu}^{13}+\mu_{vu}^{14})w_u+(-\mu_{uv}^{9}+\mu_{uv}^{10}+\mu_{uv}^{11}-\mu_{uv}^{12}+\mu_{uv}^{13}-\mu_{uv}^{14}+\mu_{uv}^{3}-\mu_{uv}^{5})w_u\right]\right\}$$

$$=\sum_{u\in V}\left\{E_{idle}+E_{sleep}+\sum_{v\in V}\left[(-\mu_{vu}^{13}+\mu_{vu}^{14}-\mu_{uv}^{9}+\mu_{uv}^{10}+\mu_{uv}^{11}-\mu_{uv}^{12}+\mu_{uv}^{13}-\mu_{uv}^{14}+\mu_{uv}^{3}-\mu_{uv}^{5})\right]\right\}w_u$$

Therefore, (SUB 3.4) can be further decomposed into $|V|$ independent subproblems. For each node *v*,

$$\min\left\{E_{idle}+\sum_{v\in V}\left[(-\mu_{vu}^{10}+\mu_{vu}^{11}-\mu_{uv}^{6}+\mu_{uv}^{7}+\mu_{uv}^{8}-\mu_{uv}^{9}+\mu_{uv}^{10}-\mu_{uv}^{11}+\mu_{uv}^{14}-\mu_{uv}^{16})\right]\right\}w_u \tag{SUB 3.5.1}$$

Subject to:

$$w_u \in \{0,1,2,...,T\}$$

For each (SUB 3.5.1), we calculate the coefficient

$E_{idle}+\sum_{v\in V}\left[(-\mu_{vu}^{10}+\mu_{vu}^{11}-\mu_{uv}^{6}+\mu_{uv}^{7}+\mu_{uv}^{8}-\mu_{uv}^{9}+\mu_{uv}^{10}-\mu_{uv}^{11}+\mu_{uv}^{14}-\mu_{uv}^{16})\right]$ of each node *u*. If the coefficient of node *u* is negative, then set w_u to be *T*, otherwise 0. The computational complexity of (SUB 3.5.1) is $O(1)$ for each node *u*.

3.2.6 Subproblem 6 (related to decision variable r_u)

$$\min \sum_{u\in V}\sum_{v\in V}\left[\left(\mu_{uv}^{6}-\mu_{uv}^{7}\right)r_u+V\cdot e(r_u)\right] \qquad \text{(SUB 3.6)}$$

Subject to:

$$r_u \in R \qquad \forall u \in V \qquad (3.15)$$

$$r_s \neq 0 \qquad \forall s \in S\,. \qquad (3.16)$$

(SUB 3.6) can be further decomposed into $|V|$ independent subproblems. For each node *u*,

$$\min \sum_{v\in V}\left(\mu_{uv}^{6}-\mu_{uv}^{7}\right)r_u+V\cdot e(r_u) \qquad \text{(SUB 3.6.1)}$$

Subject to:

$$r_u \in R$$

$$r_s \neq 0$$

For each (SUB 3.6.1) subproblem, if $\left(\mu_{uv}^{6}-\mu_{uv}^{7}\right)\geq 0$, set r_u to be zero, else set r_u to be $\dfrac{\left(\mu_{uv}^{7}-\mu_{uv}^{6}\right)}{2\cdot V}$. The computational complexity of (SUB 3.6.1) is $O(1)$ for each node *v*.

3.2.7 Subproblem 7 (related to decision variable ϕ_{uv})

$$\min \sum_{u \in V} \sum_{v \in V} \left(-\mu_{uv}^{6} M + \mu_{uv}^{7} \eta_{uv} - \mu_{uv}^{8} + \mu_{uv}^{16} \right) \phi_{uv} \qquad \text{(SUB 3.7)}$$

Subject to:

$$\phi_{uv} = 0 \ or \ 1 \qquad \forall u, v \in V. \qquad (3.17)$$

(SUB 3.7) can be further decomposed into $|V \times V|$ independent subproblems. For each node v,

$$\min \left(-\mu_{uv}^{6} M + \mu_{uv}^{7} \eta_{uv} - \mu_{uv}^{8} + \mu_{uv}^{16} \right) \phi_{uv} \qquad \text{(SUB 3.7.1)}$$

Subject to:

$$\phi_{uv} = 0 \ or \ 1$$

For each (SUB 3.7.1), we calculate the coefficient $-\mu_{uv}^{6} M + \mu_{uv}^{7} \eta_{uv} - \mu_{uv}^{8} + \mu_{uv}^{16}$ of each link (u, v). If the coefficient is negative, then set ϕ_{uv} to be 1, otherwise 0. The computational complexity of (SUB 3.7.1) is $O(1)$ for each link (u, v).

3.2.8 Subproblem 8 (related to decision variable z_{uv})

$$\min \sum_{u \in V} \sum_{v \in V} \left\{ (-\mu_{uv}^{15} - \mu_{uv}^{16}) z_{uv} \right\} \qquad \text{(SUB 3.8)}$$

Subject to:

$$\sum_{u \in V} z_{(uv)} \le 1 \qquad \forall v \in V \qquad (3.6)$$

$$z_{uv} = 0 \ or \ 1 \qquad \forall u, v \in V. \qquad (3.14)$$

(SUB 3.8) can be further decomposed into $|V \times V|$ independent subproblems. For each node link (u, v),

$$\min\,(-\mu_{uv}^{15}-\mu_{uv}^{16})z_{uv} \qquad \text{(SUB 3.8.1)}$$

Subject to:

$$\sum_{u\in V} z_{(uv)} \le 1$$

$$z_{uv} = 0 \;\; or \;\; 1$$

The proposed algorithm for solving (SUB 3.8.1) is described as follows:

Step 1: For each link (u,v) compute the coefficient $-\mu_{uv}^{15}-\mu_{uv}^{16}$ for each z_{uv}.

Step 2: For all incoming links of node v, find the smallest coefficient. If the smallest coefficient is negative then set the corresponding z_{uv} to be 1 and the other incoming links z_{uv} to be 0, otherwise set all incoming link z_{uv} to be 0. Repeat step 2 for all nodes. The computational complexity of (SUB 3.8.1) is $O(V)$ for each node v.

3.2.9 Subproblem 9 (related to decision variable z_{uv1})

$$\min \sum_{u\in V}\sum_{v\in V}\left\{(-\mu_{uv}^{9}M+\mu_{uv}^{10}M+\mu_{uv}^{16})z_{uv1}\right\} \qquad \text{(SUB 3.9)}$$

Subject to:

$$z_{uv1} = 0 \;\; or \;\; 1 \qquad \forall u,v \in V. \qquad (3.12)$$

(SUB 3.9) can be further decomposed into $|V\times V|$ independent subproblems. For each node link (u, v),

$$\min\,(-\mu_{uv}^{9}M+\mu_{uv}^{10}M+\mu_{uv}^{16})z_{uv1} \qquad \text{(SUB 3.9.1)}$$

Subject to:

$$z_{uv1} = 0 \;\; or \;\; 1$$

For each (SUB 3.9.1), we calculate the coefficient $(-\mu_{uv}^{9}M+\mu_{uv}^{10}M+\mu_{uv}^{16})$ of each link (u, v). If the coefficient is negative, then set z_{uv1} to be 1, otherwise 0. The computational complexity of (SUB 3.9.1) is $O(1)$ for each link (u, v).

3.2.10 Subproblem 10 (related to decision variable z_{uv2})

$$\min \sum_{u\in V}\sum_{v\in V}\left\{(-\mu_{uv}^{11}M+\mu_{uv}^{12}M+\mu_{uv}^{16})z_{uv2}\right\} \quad \text{(SUB 3.10)}$$

Subject to:

$$z_{uv2} = 0 \;\; or \;\; 1 \qquad \forall u,v\in V. \quad (3.13)$$

(SUB 3.10) can be further decomposed into $|V\times V|$ independent subproblems. For each node link (u, v),

$$\min (-\mu_{uv}^{11}M+\mu_{uv}^{12}M+\mu_{uv}^{16})z_{uv2} \quad \text{(SUB 3.10.1)}$$

Subject to:

$$z_{uv2} = 0 \;\; or \;\; 1$$

For each (SUB 3.10.1), we calculate the coefficient $(-\mu_{uv}^{11}M+\mu_{uv}^{12}M+\mu_{uv}^{16})$ of each link (u, v). If the coefficient is negative, then set z_{uv2} to be 1, otherwise 0. The computational complexity of (SUB 3.10.1) is $O(1)$ for each link (u, v).

3.2.11 Subproblem 11 (related to decision variable D_{uv})

$$\min \sum_{u\in V}\sum_{v\in V}\left\{(-\mu_{uv}^{13}-\mu_{uv}^{14}-\mu_{uv}^{15})D_{uv}\right\} \quad \text{(SUB 3.11)}$$

Subject to:

$$D_{uv}\in\{0,1,2,\ldots,T\} \qquad \forall u,v\in V. \quad (3.18)$$

(SUB 3.11) can be further decomposed into $|V\times V|$ independent subproblems. For each node link (u, v),

$$\min (-\mu_{uv}^{13}-\mu_{uv}^{14}-\mu_{uv}^{15})D_{uv} \quad \text{(SUB 3.11.1)}$$

Subject to:

$$D_{uv} \in \{0,1,2,...,T\}$$

For each (SUB 3.11.1), we calculate the coefficient $(-\mu_{uv}^{13} - \mu_{uv}^{14} - \mu_{uv}^{15})$ of each link (u, v). If the coefficient is negative, then set D_{uv} to be T, otherwise 0. The computational complexity of (SUB 3.11.1) is $O(1)$ for each link (u, v).

3.3 The Dual Problem and the Subgradient Method

According to the algorithms proposed above, we could effectively solve the Lagrangean relaxation problem optimally. Based on the weak Lagrangean duality theorem (for any given set of nonnegative multipliers, the optimal objective function value of the corresponding Lagrangean relaxation problem is a lower bound on the optimal objective function value of the primal problem[7]), $Z_{LR}\left(\mu^1,\mu^2,...,\mu^{16}\right)$ is a lower bound on Z_{IP}. We construct the following dual problem to calculate the tightest lower bound and solve the dual problem by using the subgradient method.

Dual Problem

$Z_{LR}=\min Z_{LR}\left(\mu^1,\mu^2,...,\mu^{16}\right)$

Subject to:

$\mu^1,\mu^2,...,\mu^{16}\geq 0$

Let the vector *S* be a subgradient of $Z_{LR}\left(\mu^1,\mu^2,...,\mu^{16}\right)$ at $\left(\mu^1,\mu^2,...,\mu^{16}\right)$. In iteration *k* of the subgradient optimization procedure, the multiplier vector $m^k=\left(\mu^{1k},\mu^{2k},...,\mu^{16k}\right)$ is updated by $m^{k+1}=m^k+\alpha^k S^k$,

where

$s^k\left(\mu^1,\mu^2,...,\mu^{16}\right)=$

The step size α^k is determined by $\delta\dfrac{Z_{IP}^k-Z_{LR}\left(m^k\right)}{\left\|S^k\right\|^2}$, where Z_{IP}^k is the best primal objective function value found by iteration *k* (an upper bound on the optimal primal objective function value), and δ is a constant $\left(0 \le \delta \le 2\right)$.

Chapter 4 Getting Primal Feasilbe Solution

After optimally solving each subproblem, we derive a set of multipliers and decision variables. Nevertheless, the solution wouldn't be a feasible solution for primal problem because of the violations of the original constraints. Hence, we must to find some heuristics to tune these multipliers and decision variables to ensure that all of the primal constraints are satisfied.

In this chapter, we proposed a two-phase heuristic. Because of the difficulty of deciding all decision variables simultaneously, we divide this complex problem into two parts and obtain all decision variables respectively. The part one is used for routing procedure and the part two is used for scheduling procedure.

To separate the routing and scheduling procedure can help us determine each variable efficiently. However, this way may potentially lead to the transmission latency violate the maximum end-to-end delay requirement. In order to improve the solution quality and decrease the maximum end-to-end delay within a reasonable range, we proposed a reroute heuristic to avoid that situation.

4.1 Getting Primal Heuristic

In a data-centric routing scheme, all sources send information with non-deterministic redundancy. If the flows with redundant information aggregate at a particular node, they will be much more energy efficient than those choosing the nearest routing path directly. Hence, for the purpose of minimizing the total energy consumption of a data-centric wireless sensor networks, the aggregation of the flows coming from different sources is a better way to achieve energy efficiency.

In this problem, we have six major decision variables x_p, y_{uv}, r_u, n_u, m_u, and w_u to be determined. Once $\{x_p\}$ are determined, $\{y_{uv}\}$ and $\{r_u\}$ can be handily derived. With the determined $\{x_p\}$, $\{y_{uv}\}$ and $\{r_u\}$, we can construct a data aggregation tree easily. After the data aggregation tree has been constructed, we can much more easier to decide $\{n_u\}$, $\{m_u\}$, and $\{w_u\}$.

Therefore, in order to solving this problem efficiently, we divide the getting primal feasible procedure into two phases to obtain all decision variables respectively. Heuristic phase one is used for routing policy and phase two is used for scheduling policy.

4.1.1 Heuristic for Routing Policy

In the beginning of our algorithm, we set the arc weight of link (u,v) to be $u_{suv}^1 + u_{uv}^2 + e(\frac{\eta_{uv}}{MaxRadius})$, and run Dijkstra's algorithm to get the shortest path and its cost from each source node to sink node.

$e(\frac{\eta_{uv}}{MaxRadius})$ stands for the energy cost from node u to v, which is a exponential function of the distance between node u and node v. The reason why we divide it by $MaxRadius$ is for normalization purpose such that the arc weight will not be dominated by η_{uv}.

After obtained the shortest path and its cost of each source, we selected a path with the minimum cost and set the corresponding y_{uv} to be one and adjust the arc weights on this path to be zero. Then run Dijkstra's algorithm once again.Repeat adjust arc weights procedure and Dijkstra's algorithm until there is a path from each source to sink.

The basic idea of phase two is that if a path has been involved in the data aggregation tree, the other nodes should try to connect to this path as possible as they can. Because when a path has been involved into the tree, all nodes among this path had set their transmission radii already. Hence, the cost of this path should not be the original arc weight but zero. In the following, we show the detail procedures in Table 4-1.

Table 4-1 Phase 1 – Routing Policy

Step 1-1. Set the arc weight for link (u,v) to be $u^1_{suv} + u^2_{uv} + e(\frac{\eta_{uv}}{MaxRadius})$, and run Dijkstra's algorithm to get the shortest path from each source node.
Step 1-2. Choose a path with the smallest cost, set each value of $\{y_{uv}\}$ to be one, if this y_{uv} is on the corresponding path, and adjust the arc weight of these links from $u^1_{suv} + u^2_{uv} + e(\frac{\eta_{uv}}{MaxRadius})$ to be zero.
Step 1-3. Repeat Step 1-1 ~ 1-2 until all source have a path to sink node.
Step 1-4. Set each value of $\{r_u\}$ to be the nearest value from $\{R\}$, if the corresponding y_{uv} is one.
Step 1-5. Using $\{y_{uv}\}$ to construct a data aggregation tree.

4.1.2 Heuristic for Scheduling Policy

The basic idea of phase two is to minimize the total number of slots used by transmission. Hence, to assign the same slot to each link as many as possible should be a good way to fulfill this goal. The out-degree of a particular node stands for the possibility of this node will potentially influence the transmission of its neighbor. If we let a node with the smaller out-degree have higher priority to send its information, there will be more nodes could transmit their data in a slot. Therefore, it may potentially reduce the total number of slots used by transmission. In the following, we show the detail procedures in Table 4-2.

Table 4-2 Phase 2 – Scheduling Policy.

Step 2-1. By running topology-sort algorithm, we can derive the outliers of the data aggregation tree. Put these outliers into *stack_number1*.
Step 2-2. Sort *stack_number1* by out-degree, and pop a node with the smallest out-degree. If this node doesn't interfere the transmissions of the previous slot, do Step 2-3, else put this node into *stack_number2*.
Step 2-3. Set each value of $\{w_u\}$ to be *current_slot*, each value of $\{n_u\}$ to be the minimum m_u of its children, and each value of $\{m_u\}$ to be the maximum w_u of its children. Repeat Step 2-1 ~ 2-2 until *stack_number1 is empty*.

Step 2-4. Swap the values of *stack_number1* and *stack_number2* and set *current_slot* to be *current_slot* plus one

Step 2-5. Repeat Step 2-1 ~ 2-4 until both *stack_number1* and *stack_number2* are empty.

4.2 Rerouting Heuristic

By our two phases algorithm, we can find a good solution with low-energy cost. Sometimes, however, the latency of this solution may violate the maximum end-to-end delay requirement. In order to improve the solution quality and decrease the maximum end-to-end delay within a reasonable range, we proposed a reroute heuristic to avoid that situation.

In the following, we show the detail procedures in Table 4-3.

Table 4-3 Rerouting Heuristic

Step 1:Identify the path (denoted as *P*) that incurs the highest end-to-end delay.
Step 2: Investigate nodes located on *P* one by one. For each checked node (denoted as *n*) , examine each node (denoted as *k*) If the end-to-end delay of node *n* plus one unit of delay is smaller than that of *k*, then reroute the path from *n* to *k*.
Step 3: Update the corresponding decision variables and reconstruct data aggregation tree.
Step 4: Repeat Step 1 – 3, until m_{sink} within the Maximum end-to-end delay requirement.

4.3 Lagrangean Relaxation Based Algorithm

We have already described the Lagrangean Relaxation process in Chapter 3.The complete Lagrangean Relaxation based algorithm is described as follows:

Table 4-4 Lagrangean Relaxation Based Algorithm

begin

    Initialize the Lagrangean multiplier vector ($\mu^1, \mu^2, ..., \mu^{16}$) to be all zero vectors;

    UB := 0; LB:= - Iinfinity;

    improve_counter := 0;

    step_size_cofficient := 2;

    for iteration := 1 to Max_Iteration_Number do

    begin

        run subproblem1 (Sub1)

        run subproblem2 (Sub2)

        run subproblem3 (Sub3)

        run subproblem4 (Sub4)

        run subproblem5 (Sub5)

        run subproblem6 (Sub6)

        run subproblem7 (Sub7)

        run subproblem8 (Sub8)

        run subproblem9 (Sub9)

        run subproblem10 (Sub10)

        run subproblem11 (Sub11)

```
        calculate Zdu;
        if Zdu> LB then
            LB := Zdu; improve_counter +1;
            improve_counter := 0;  δ  :=  δ / 2 ;
        run Primal_Heuristic_Algortihm;
        if Zdu> Max_Delay_Requirement then
            run Rerouting Heuristic;
        calculate Zip;
        if Zip < UB then UB = Zip;
        run update-step-size;
        run update-Lagrangean-multiplier;
    end;
end;
```

Chapter 5 Computational Experiments

In this chapter, in order to test the quality of our getting primal feasible solution, we conduct several computational experiments. In the mean time, for the purpose of evaluating the solution quality , we implement two simpe algorithm - Shortest Path Tree(SPT) and Center Nearest Source(CNS) [13] for comparison.

5.1 Experiment Environment

The computational experiments program has been written in C and using a Pentium IV 2.8GHz, 1024MB, Windows 2000 Server Pack4 enviorment. Table 5-1 shows the general parameters and test platform for the experiments

Table 5-1 Experiment Environment and Parameters

Experiment Environment and Parameters	
Parameter	**Value**
Number of Nodes	10 ~ 100
Density of Sources	20% ~ 40%
Number of Iterations	2000
Improvement Counter	30
Initial Upper Bound	Solution of 1st Getting Primal Feasible
Initial Upper Multiplier	0
Initial Scalar of step size	2
Test Platform	CPU : Intel(R) Pentium-IV 2.8GHz
	RAM : 1024 MB
	OS : Windows 2000 SP 4

5.2 Simple Algorithms and Metrics

According to [13], for the routing problem, we implement Shortest Path Tree (SPT) Algorithm as Simple Algorithm 1 (SA1), and Center Nearest Source (CNS) Algorithm as Simple Algorithm 2(SA2). In addition, for the scheduling problem, we use the same scheduling policy as LR in order to conserve the righteousness for comparison.

We denote our dual solution as Zdu, and Lagrangean-based Heuristic as LR. Besides, we use two metrics – "Gap" and "Improvement Ratio" to evaluate our solution quality. Gap is calculated by $\frac{LR-LB}{LB}\times 100\%$. And Improvement Ratio is calculated by $\frac{SA-LR}{LR}\times 100\%$.

5.3 Experiment Scenarios

In order to test the solution quality of our algorithm, we design several scenarios with different feature.

1. Random Network with Different Number of Sensor Nodes
2. Grid Network with Different Number of Sensor Nodes
3. Random Network with Different Density of Source nodes

In the following sections, we conduct a number of experiments in the random and grid network. The major difference between random network and grid network is the sensor placement manner. In a random network, all sensors are scattered disorderly in the field. Both the position and density of the sensor nodes are haphazard. In a grid network, sensors are placed in uniformly. Generally Speaking, we divide the network topology into a number of blocks. Each sensor node will be placed on a block without any sensor. Therefore, both the position and density of the sensors will be uniform.

Furthermore, two different sensor placement manners, namely *random source*s and *congregated sources* are tested in the random network. In the *random sources* manner, the sink node will be placed in the center of the topology, and all sources will be dispersedly scattered around the sink. In the *congregated sources* manner, the sink node will be placed in the corner of the topology, and all sources will be scattered in another corner.

In the last part, in order to observe the impact effect when the number of sources is growing up, we carry a experiment out in random network with different density of sources.

In Section 5.4, 5.5, and 5.6, we use several tables and diagrams to explain our experiment result and solution quality. In Section 5.7, we will make a short discussion about our research.

5.4 Random Network with Different Number of Sensor Nodes

5.4.1 Random Network with Different Number of Sensor Nodes (Random Source)

Table 5-2 Experiment Result of Random Network with Different Number of Sensors (Random Source)

Number of Sensor Nodes	Lower Bound (LB)	Upper Bound (UB)	Gap(%)	Simple Algorithm1 (SA1)	Imp Ratio to SA1	Simple Algorithm1 (SA2)	Imp Ratio to SA2
10	69.9167	75.82	8.4433333	87.61	15.549987	84.26	11.131628
20	111.2325	123.6200	11.1366	137.56	11.2765	235.3	90.3414
30	174.3990	207.9700	19.2495	303.14	45.7614	372.87	79.2903
40	432.4178	518.3800	19.8794	691.08	33.3153	799.86	54.2999
50	491.2730	660.1500	34.3754	868.9	31.6216	837.01	26.7909
60	675.7940	923.7100	36.6851	1400.72	51.6407	1422.76	54.0267
70	976.4596	1222.4200	25.1890	1660.97	35.8756	1578.16	29.1013
80	1077.4176	1423.8700	32.1558	1781.28	25.1013	1818.49	27.7146
90	1046.8646	1526.5100	24.8258	1987.74	30.2147	1886.98	23.6140
100	1328.6539	1702.1900	28.1139	2234.32	31.2615	2390.35	40.4279

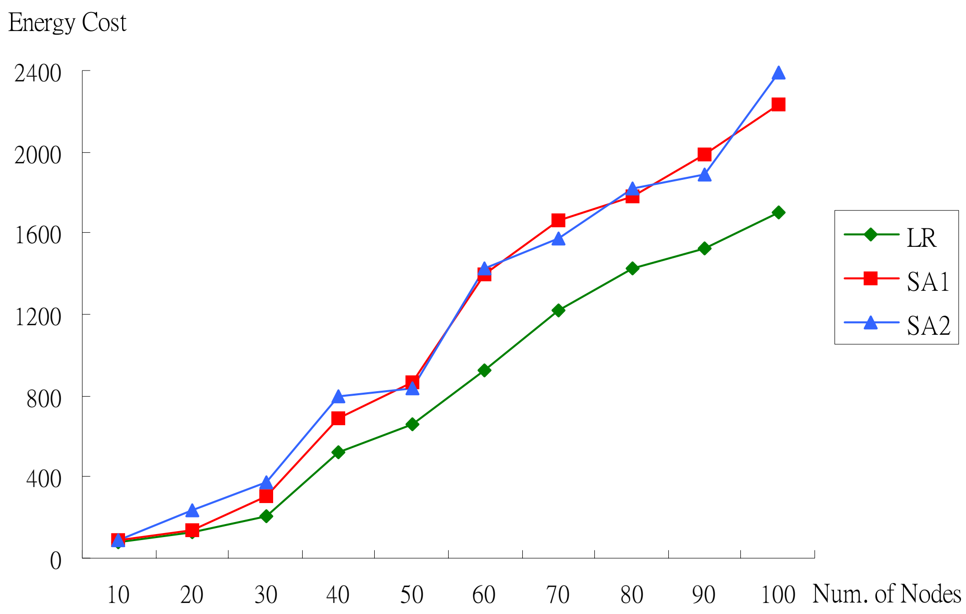

Fig 5-1 Energy Cost of Random Network with Different Number of Sensor Nodes (Random Source)

5.4.2 Random Network with Different Number of Sensor Nodes (congregated source)

Table 5-3 Experiment Result of Random Network with Different Number of Sensors (congregated Source)

Number of Sensor nodes	Lower Bound (LB)	Upper Bound (UB)	Gap(%)	Simple Algorithm1 (SA1)	Imp Ratio to SA1	Simple Algorithm1 (SA2)	Imp Ratio to SA2
10	67.239148	67.7933333	0.8242004	90.953333	34.162651	86.01	26.870882
20	117.37223	120.8675	2.9779361	148.9825	23.261009	139.8675	15.719693
30	229.29795	235.456667	2.6859013	315.52	34.003426	298.49	26.770673
40	351.55937	411.193333	16.962701	615.86333	49.77464	522.47667	27.063506
50	509.00803	625.366667	22.859882	840.53667	34.407015	737.02667	17.855125
60	707.83128	824.2275	16.444063	1094.1875	32.753093	984.4625	19.440628
70	847.95089	955.155	12.642726	1260.715	31.990619	1162.855	21.745162
80	835.77045	1087.69667	30.142992	1510.21	38.844776	1237.8	13.80011
90	961.52129	1236.21333	28.568483	1724.6033	39.506935	1584.8833	28.204679
100	1188.3199	1455.31667	22.468425	2088.5333	43.510576	1769.24	21.570792

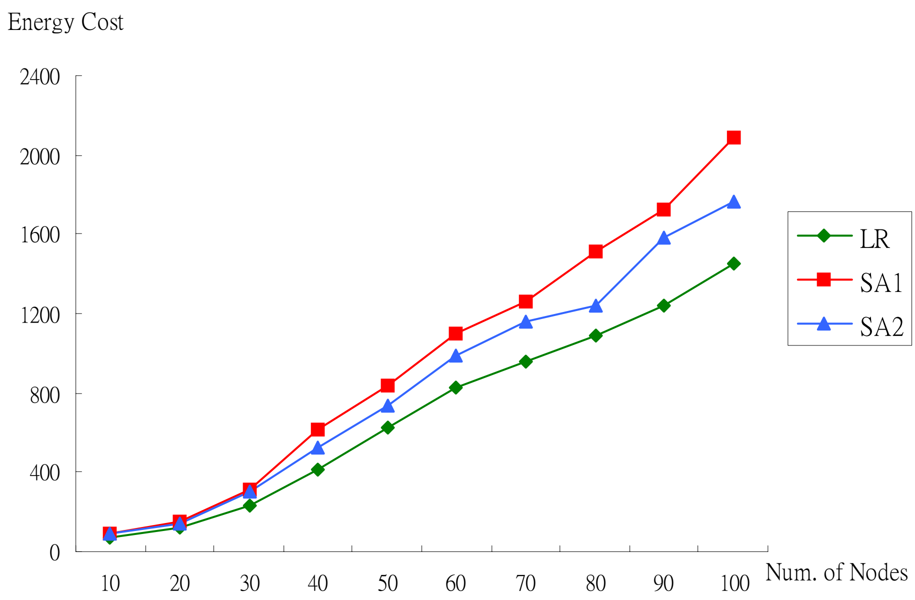

Fig 5-2 Energy Cost of Random Network with Different Number of Sensor Nodes (congregated sources)

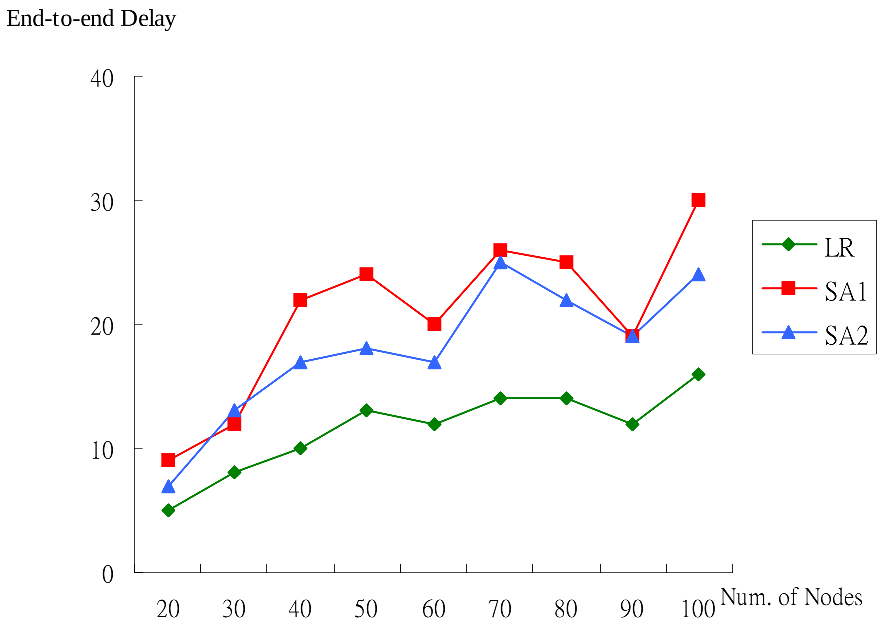

Fig 5-3 End-to-End Delay of Random Network

with Different Number of Sensor Nodes

5.5 Random Network with Different Number of Sensor Nodes

Table 5-4 Experiment Result of Grid Network with Different Number of Sensors

Number of Sensor nodes	Lower Bound (LB)	Upper Bound (UB)	Gap(%)	Simple Algorithm1 (SA1)	Imp Ratio to SA1	Simple Algorithm1 (SA2)	Imp Ratio to SA2
16	79.1978	85.4300	7.87	119.93	40.3839	128.41	50.3102
25	102.2101	119.1800	16.60	137.22	15.1368	153.52	28.8136
36	249.9213	299.0800	19.67	401.74	34.3253	396.07	32.4295
49	456.2145	604.8900	32.59	808.26	33.6210	725.23	19.8945
64	658.8879	974.5100	47.90	1502.75	54.2057	1722.12	76.7165
81	1109.5546	1358.3800	22.43	2520.29	85.5364	2276.96	67.6232
100	1069.9521	1527.8100	42.79	2789.07	82.5535	2448.57	60.2667

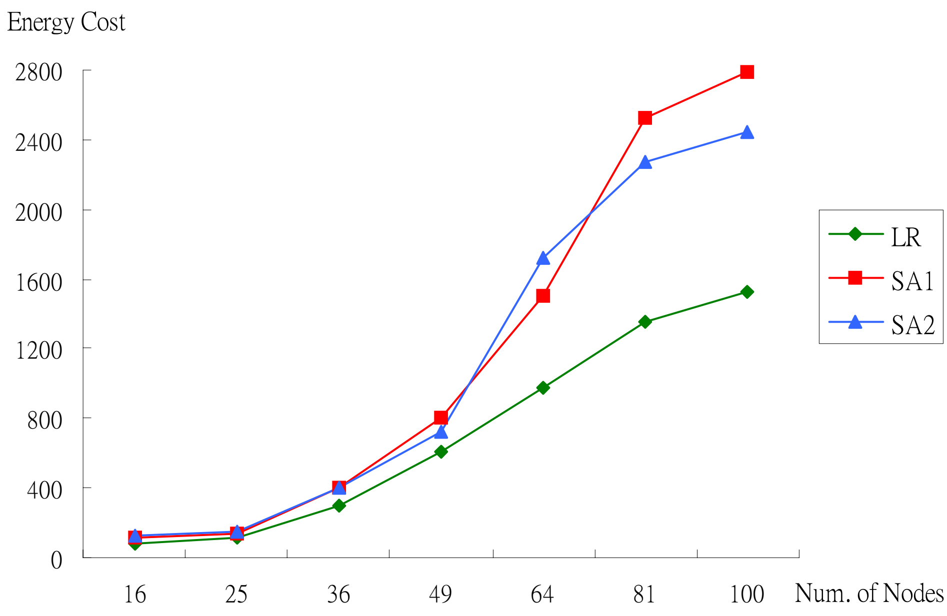

Fig 5-4 Energy Cost of Grid Network with Different Number of Sensor Nodes

5.6 Random Network with different density of source nodes

Table 5-5 Experiment Result of Random Network with Different Number of Sources

Number of Sensor nodes	LR	Simple Algorithm1 (SA1)	Imp Ratio to SA1	Simple Algorithm1 (SA2)	Imp Ratio to SA2
10	959.67	1201.07	25.1545	1067.4	11.2257
15	1175.68	1551.52	31.9679	1301.99	10.7436
20	1332.12	1816.37	36.3518	1539.83	15.5924
25	1386.69	1996.51	43.9767	1716.93	23.8150
30	1465.1	2168.12	47.9844	1774.57	21.1228
35	1541.68	2293.21	48.7475	1878.93	21.8755
40	1690.71	2435.66	44.0614	2021.11	19.5421

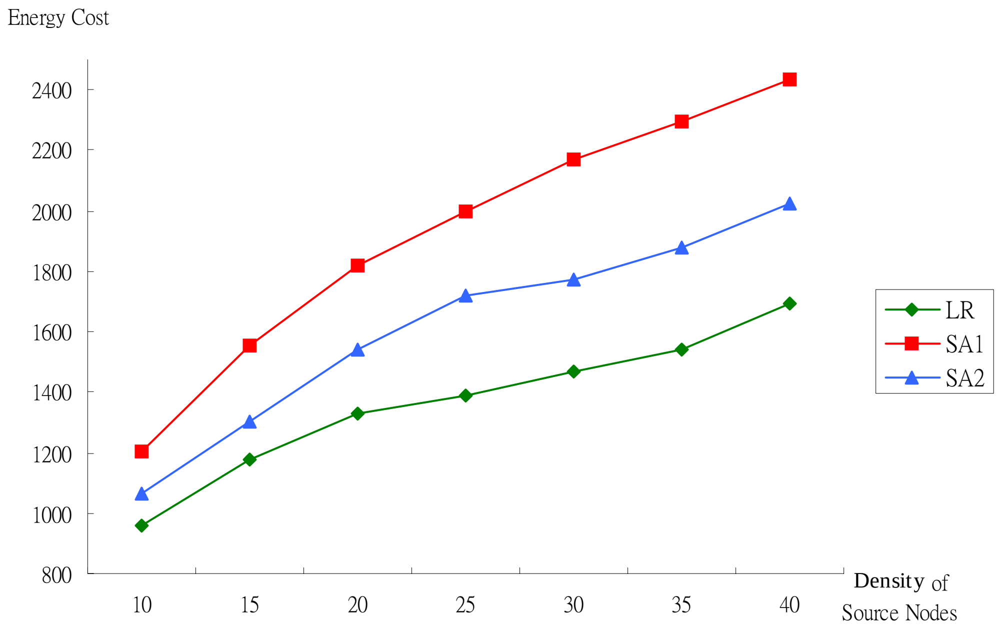

Fig 5-5 Energy Cost of Random Network with Different Density of Source Nodes

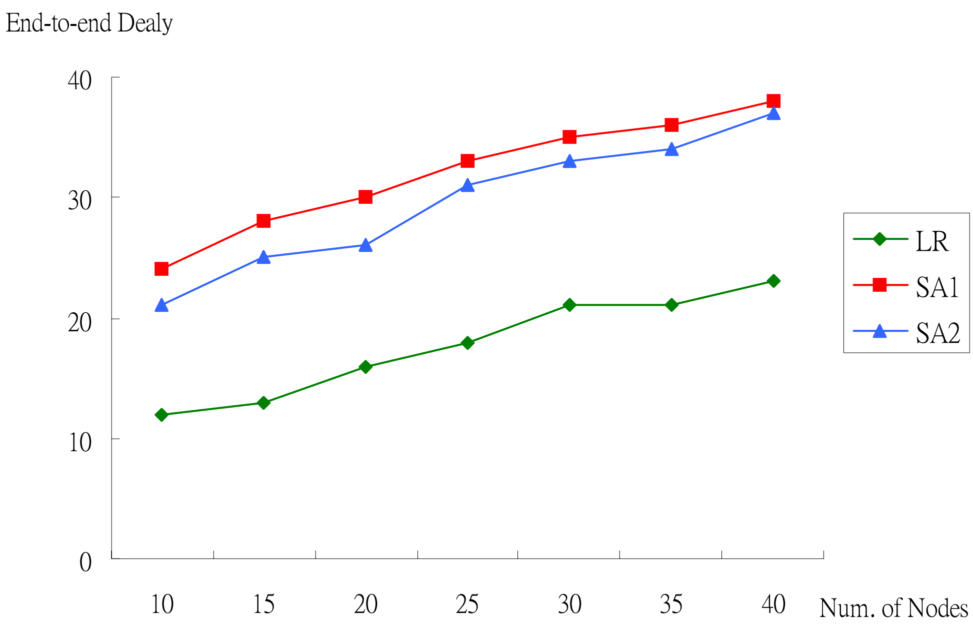

Fig 5-6 End-to-End Delay of Random Network

with Different Density of Source Nodes

5.7 Experiment Discussion

5.7.1 Topology and Sensor Placement Manner

During most of the time, as shown in Fig 5-1, we can observe that CNS (SA2) is as good as SPT (SA1). Sometimes, CNS is worse than SPT. In the *random source* model, because all sources are dispersedly scattered around the sink, CNS will potentially waste its energy on transmitting the information to a remote source. That's the reason why SPT is better than CNS in Fig 5-1. However, our LR-based algorithm will not lead to the side effect like CNS. Therefore, our LR-based algorithm is eminently superior to these simple algorithms

In the *congregated source* model, sink is located in the corner of the topology. And all sources are placed in another side. Hence, CNS will aggregate all information from each source quickly, and deliver these messages through the nearest source to the sink. As shown in Fig 5-2, CNS is much better than SPT in this placement manner. Nevertheless, the algorithm we proposed still outperforms SPT and CNS significantly

In the Fig 5-4, we can perceive that both improvement ratio and duality gap in the grid network are larger than those in the random network. For these curious results, we will discuss that further in the Section 5.7.3.

5.7.2 Density of Sources

In this Section, we would like to measure the influence when the number of sources is increasing in a *congregated source* model.

As shown in Fig 5-5, SPT is the worse among these three, due to its selection is too intuitive. As what we mentioned before, the energy cost incurs by CNS is slightly lower than that by SPT when all sources are congregated. Our algorithm is the best of them no matter when the density of sources is growing up.

5.7.3The Sequence of Path Selection

According to our experiment results, we found something interesting. As shown in Fig 5-6, there are 2 sources A and B in a grid network. Both the node A and B possess the same amount of tranmission cost from themself to the sink. (A->C->S and B->S). The green path incurs 4 units of transmission cost., and the red path only results in 3 unis of transmission cost. Briefly, during the construction of data aggregation tree,different sequence of the path selection will lead to different results.

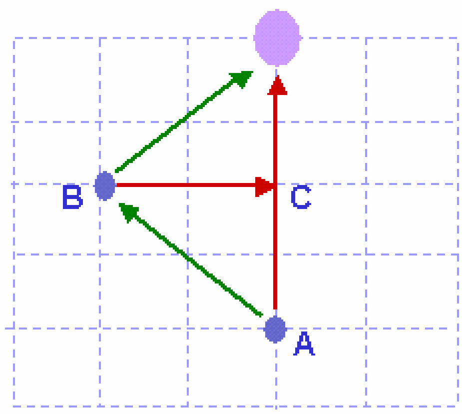

Fig 5-6 End-to-End Delay of Random Network

with Different Density of Source Nodes

Furthmore, we perceived this effect will appear more frequently in the grid network. Due to the density and position of a gird network are uniform, there are at least 2 paths from a particular node to the sink node. That's the reason why our computational experiments in the gird network will incur better improvement ratio and larger gap than those in the random network.

Chapter 6 Conclusion and Future Work

6.1 Summary

In this thesis, we emphasize on a problem of routing and scheduling the activities of all sensors in a data-centric wireless sensor networks. In chapter 2, we formulate this problem as an integer programming problem, where the objective function value is to minimize the total energy consumption, including sending, receiving, idling, and sleeping. In chapter 3 and 4, we develop a Lagrangean Relaxation based heuristic to solve this problem. Furthermore, we propose a reroute heuristic to ensure the end-to-end delay will within a reasonable range. In chapter 5, we carry a number of experiments in order to evaluate the solution quality of our approach. As shown in the diagrams of chapter 5, our algorithm significantly outperforms SPT and CNS in both random and grid network.

The contribution of this thesis can be as follows:

1. We propose a mathmatical formulation and a optimization based algorithm with jointly considering energy-efficient, delay-efficient, scheduling, and routing in the data-centric network.
2. Our Lagrangean Relaxation based solutions have significant improvement than other intentional algorithms.

6.2 Future Work

Although we take both routing and scheduling issues into consideration, there are still several progressive researches to be addressed.

As show in Fig 5-6, during the construction of data aggregation tree,different sequence of the paths will lead to different results. The data-centric routing problem is the the well-known Steiner tree problem which is proven to be a NP-Hard problem. The optimal solution may the permutation of these paths from each source to sink.By Lagrangean Relaxation Method , we can derive a number of multipliers and decision variables. Both these multipliers and dual decision variables are the good hint to build a primal heuristic. Through various experiments, we find the current getting primal heuristic. Nevertheless, we cant presume that the current combination of mulipliers is the best. There may be others combination of multipliers that will bring a better solution.

In this thesis, our objective function is to minimize the total energy consumption including sending, receiveing, idling, and sleeping. It implies that the lifetime of this network will be prolonged. However, we only considered the activities of all sensors in a single run. After a long period, some sensors in the data aggregation tree may perish due to the heavy traffic. Therefore, taking some issues such as load balancing or .the permutation of aggregation tree into consideration will make this research be much more worthy.

References

[1] W. Ye, J. Heidemann, and D. Estrin, “An energy-efficient MAC protocol for wireless sensor networks”, *In 21st Conference of the IEEE Computer and Communications Societies (INFOCOM*), volume 3, pages 1567(1576, June 2002)

[2] W. Ye, J. Heidemann, and D. Estrin, “Medium Access Control with Coordinated, Adaptive Sleeping for Wireless Sensor Networks”, *Technical Report USC ISI-TR-567, (Accepted to appear in ACM/IEEE Transactions on Networking 2003.)*

[3] G. Lu, B. Krishnamachari, and C. Raghavendra, “An Adaptive Energy-Efficient and Low-Latency MAC for Data Gathering in Sensor Networks”, *in 4th IEEE International Workshop on Algorithms for Wireless, Mobile, Ad Hoc and Sensor Networks WMAN 2004.*

[4] T. Dam, and K. Langendoen, “An Adaptive Energy-Efficient MAC Protocol for Wireless Sensor Networks”, *SenSys*, 2003, Los Angeles, California.

[5] G. Lu, B. Krishnamachari, and C. Raghavendra, “Delay Efficient Sleep Scheduling in Wireless Sensor Networks”, *in 21st Conference of the IEEE Computer and Communications Societies. (INFOCOM 2004)*

[6] S. Begum, S. C. Wang, B. Krishnamachari, and A. Helmy, “Energy-efficient and low-latency scheduling technique for wireless sensor networks”, *In 21st Conference of the IEEE Computer and Communications Societies. (INFOCOM 2004)*

[7] M.L. Fisher, “The Lagrangean Relaxation Method for Solving Integer

Programming Problems", *Management Science*, Volume 27, Number 1, pp. 1-18, January 1981.

[8] M.L. Fisher, "An Application Oriented Guide to Lagrangian Relaxation", *Interfaces*, Volume 15, Number 2, pp. 10-21, April 1985.

[9] A.M. Geoffrion, "Lagrangean Relaxation and its Use in Integer Programming", *Mathematical Programming Study*, Volume 2, pp. 82-114, 1974.

[10] M.S. Bazaraa, H.D. Sherali, and C.M. Shetty, "Lagrangian Duality and Saddle Point Optimality Conditions", *Nonlinear Programming: Theory and Algorithms*, 2nd Edition, pp. 199-242, John Wiley & Sons, Inc, Singapore, 1993.

[11] J. Heidemann, F. Silva, C. Intanagonwiwat, R. Govidan, and D. Estrin "Building Efficient Wireless Sensor Networks with Low Level Naming", *18th ACM Symposium on Operating System Principles*, October 21-24, 2001 .

[12] C. Intanagonwiwat, R. Govidan, and D. Estrin, "Directed Diffusion: A Scalable and Robust communication paradigm for Sensor Networks ", *ACM/IEEE international conference on Mobile Computing and Networks (MobiCom 2002)*, August 2000, Boston, Massachusetts

[13] B. Krishnamachari, D. Estrin, and S. Wicker, "Modeling Data-Centric Routing in Wireless Sensor Networks", *IEEE INFOCOM 2002*.

[14] H. S. Yen, F.Y.S. Lin, and S.P. Lin, "Efficient Data-centric Routing

in Wireless Sensor Networks", *IEEE ICC 2005*.

[15] J. N. Karaki and A. E. Kamal, “Routing Techniques In Wireless Sensor Network: A Survey”, *IEEE Wireless Communication* Dec 2004.

[16] 1.I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, “Wireless sensor network: a survey”, *Computer Network, vol. 38, pp. 393-422,* 2002

簡　歷

姓　名：王弘翕

出生地：臺灣省臺北市

出生日：中華民國 七十年 十月十八日

學　歷：九十一年九月至九十三年六月

國立臺灣科技大學資訊管理學系

九十三年九月至九十五年七月

國立臺灣大學資訊管理學研究所