國立交通大學

網路工程研究所

碩 士 論 文

在無線感測網路中以 Quorum 為基礎處理多個查詢的省電繞徑演算法

Quorum-based Energy Efficient Routing for Continuous Queries in a Wireless Sensor Network

研 究 生：徐子文

指導教授：曾煜棋 教授

邵家健 教授

中 華 民 國 九 十 七 年 六 月

在無線感測網路中以 Quorum 為基礎
處理多個查詢的省電繞徑演算法
Quorum-based Energy Efficient Routing for Continuous Queries in a Wireless Sensor Network

研 究 生：徐子文　　Student：Tsu-Wen Hsu

指導教授：曾煜棋 教授　　Advisor：Prof. Yu-Chee Tseng

邵家健 教授　　Prof. John Kar-Kin Zao

國 立 交 通 大 學

網 路 工 程 研 究 所

碩 士 論 文

A Thesis

Submitted to Institute of Network Engineering

College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Computer Science

June 2008

Hsinchu, Taiwan, Republic of China

中華民國九十七年六月

Quorum-based Energy Efficient Routing for Continuous Queries in a Wireless Sensor Network

Student: **Tsu-Wen Hsu**

Advisor: **Prof. Yu-Chee Tseng**

Co-advisor: **Prof. John Kar-Kin Zao**

Department of Computer Science

National Chiao-Tung University

1001 Ta Hsueh Road, Hsinchu, Taiwan 300, R.O.C.

在無線感測網路中以 Quorum 為基礎
處理多個查詢的省電繞徑演算法

學生：徐子文　　　　　　　　　　　　　　　　指導教授：曾煜棋 教授
邵家健 教授

國立交通大學資訊工程學系(研究所）碩士班

摘　　　要

隨著無線感測網路相關的研究議題得到許多研究單位及學者的關注，近年來許多適用於無線感測網路，以省電為目標的 MAC 層與網路路由層通訊協定相繼發表，但現有的研究成果，絕大多數皆假設感測網路的節點具備有時間同步的條件，才能顯示其省電的效果；然而，節點們彼此要具備時間同步本身又是另一個相當耗費能源的研究課題，特別是在無線感測網路這樣的多跳通訊環境，尤其困難。之前的研究已經證明了，使用 Quorum 系統的特性可以在不具有時間同步的無線行動通訊環境下達成省電的設計；因此在這篇論文中，我們提出一個以 Quorum 系統為基礎的通訊協定，在無線感測網路中處理多個持續性查詢的同時，計算出每個節點不同的醒睡排程以節省能源不必要的浪費。另外，我們也設計遶徑演算法，使節點能夠動態調整 Quorum 系統來改變本身的醒睡排程，儘可能用較低的能源消耗去配合持續性查詢的流量需求。本篇的貢獻可以分為三個方面：首先，我們提出一個涵蓋 MAC 層與網路路由層的跨層系統設計，不同於以往只針對其中一層設計的研究，本篇論文利用 Quorum 系統適切地結合 MAC 層與網路路由層，在無線感測網路中完成支援多個持續性的查詢。第二，我們說明了如何利用簡單且不需龐大代價的局部同步演算法，協調在同一路由路徑上多個節點的 Quorum 系統來達到能源保存。第三，由於一個感測節點可能遇到必須滿足不同持續性查詢的不同流量需求又要保持較低的能源消耗的情況，我們提出讓節點能夠適應性地支援多個 Quorum 系統的方法。模擬的結果也指出我們的方法的確可以得到較低的能源消耗，並且有效地完成多個持續性查詢的工作。

關鍵字：省電、Quorum、查詢、遶徑路由、無線感測網路

Quorum-based Energy Efficient Routing for Continuous Queries in a Wireless Sensor Network

Student: Tsu-Wen Hsu

Advisor: Prof. Yu-Chee Tseng

Co-advisor: Prof. John Kar-Kin Zao

Department of Computer Science

National Chiao Tung University

ABSTRACT

Recently, several energy-efficient MAC and routing protocols have been proposed for *wireless sensor networks (WSNs)*. Most existing schemes require nodes to be synchronized in time. However, synchronization is costly, especially in multi-hop communication environments. Quorum systems have been shown to be able to support power-saving design in a wireless and mobile environment without requiring time synchronization. In this work, we propose a *quorum-based* protocol to derive their wake-up schedules to support continuous queries. We also propose a routing scheme to help nodes dynamically change their quorum patterns to meet continuous queries' traffic demands at low energy costs. The contributions of this work is three-fold. First, our design is a cross-layer design covering MAC and routing layers. While most existing protocols only address one of these layers, our work nicely tailor MAC layer using quorum systems with routing layer to support continuous queries. Second, we show how to coordinate quorums of multiple nodes in a routing path for energy conservation. A simple, inexpensive, *local synchronization* among nodes is proposed. Third, we show how to adaptively support multiple quorums in a sensor node when it is passed by several query routing paths; this can effectively meet continuous queries' traffic demands while keep energy consumption low. Simulation results indicate that our scheme can effectively support

continuous queries at low energy cost.

Keywords: energy efficient, quorum, query, routing, wireless sensor network.

誌　　　　謝

這篇論文是我在研究所兩年來學習的成果，期間受到很多人的幫助與鼓勵，在此我要簡單表達我的謝意。首先要誠摯地感謝我的指導教授曾煜棋老師，不僅是引導我進入無線感測網路這個領域並且指導這篇論文，從老師身上我學到了不少如何做好研究以及論文寫作方面的技巧，相信在未來一定有莫大的幫助。同時也感謝共同指導教授邵家健教授不時的鼓勵與建議以及論文口試委員莊東穎教授、賴添煌教授以及陳志成教授，在口試時給予我寶貴的意見，讓這篇論文更加完整。

此外，我也由衷感謝同組的孟鉉學長、佳宏學長和倫武學長，在專業領域上能和他們一起研究與討論，讓我可以學習到很多東西。其中論文及演算法的不斷改進要特別感謝孟鉉學長的指導與協助，才能順利完成。另外也謝謝同在一起打拼的 HSCC 實驗室的全體成員，不論是在研究上或者其他方面都讓我受益良多，我很珍惜和大家相處這兩年的點點滴滴。

最後，感謝我的父母及家人，在我做任何決定時都給予支持與關懷，使我能夠無虞地完成我的學業。

徐子文 於

國立交通大學 網路工程研究所 碩士班

中華民國九十七年六月

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The rapid progress of wireless communication and MEMS technology has made *wireless sensor networks (WSNs)* possible. A WSN normally consists of many inexpensive wireless sensor nodes. Each node is capable of collecting, storing, processing environmental information, and communicating with neighbor nodes. Sensor nodes usually need to configure themselves automatically and support ad hoc routing. Recently, a lot of research works have been dedicated to WSNs, such as routing and transport protocols [3][8], self-organizing schemes [12][20], deployment [10] [15], localization [1][16]. Applications of WSNs include emergency guiding [13][17], lighting control [18][19], and environmental monitoring [21].

Since sensor nodes typically use batteries as their power supplies, power saving is an important issue in WSNs. Several energy-efficient MAC protocols have been proposed [6][24][25]. In SMAC [24], nodes periodically switch to sleep mode to support low duty cycle operations. Although this scheme is very energy efficient, it may incur transmission delay. In PMAC [25], sensors can adaptively determine their sleep schedules by considering neighbor traffic patterns. In RMAC [6], sensor nodes also periodically wake up and use active periods to establish routing paths. Those that are located on the path need to stay in active mode to transmit packets. The others can go to sleep. RMAC can perform better than SMAC and PMAC in terms of transmission delay. References [5][9][23] propose energy-efficient routing schemes. In GAF [23], nodes are divided into fixed square grids. Sensors switch between sleep mode and active mode periodically. GAF guarantees that there is at least one node per grid in active mode to exchange packets with nodes in neighboring grids. Span [5] adaptively elects some nodes to stay in active mode and serves as the network backbone. Other nodes periodically check with backbone nodes to see if they need to wake up. Both [5] and [23] may have some redundant sensors to stay active. In TAP [9], redundant nodes can be identified when establishing routing paths and can go to sleep mode to further conserve energy. TAP can thus induce less latency by considering traffic flows. However, those schemes all require nodes to be synchronized in time and synchronization is costly, especially in multi-hop communication environments.

Query processing is a basic operation in WSNs. There are two types of queries. The first one is *one-shot query*, where the sink only requests an instant report from one or a set of sensors regarding their readings. The other is *continuous query*, where periodical reports from sensors are requested. Existing query processing works [2][7] focus on aggregation of multiple reports to reduce the number of transmissions or packet sizes. However, existing works do not consider query processing together with the underlying MAC protocol, which address the energy conservation issue. And little attention has been paid to reducing the packet end-to-end latency while preserving the energy saving capability. In this work, we propose a co-design of query processing together with an energy-saving MAC protocol. Therefore, idle listening of sensor nodes can be significantly reduced. Our design also focuses on how to reduce the query latencies and how to support multiple continuous queries simultaneously from different sources to different sinks with different intervals and different durations.

In our work, we adopt the *quorum-based* protocols [4][11][14][22] to derive the wake-up schedules of sensor nodes. In distributed systems, a quorum is a set of identities from which one has to obtain permission to enter a critical section to ensure mutual exclusion. It is first shown in [22] that it is possible to exploit the quorum concept to design the wake-up patterns of nodes without requiring them to synchronize their clocks. In this work, we propose that sensor nodes use a default quorum to schedule their wake-up patterns. In addition, a *local* *synchronization* mechanism is designed to enforce nodes to have similar wake-up patterns as their neighbors. This would allow nodes to have more chances to exchange packets and thus can decrease query latencies. As a node is on the paths of multiple query routes, we propose a scheme to dynamically change its quorum satisfying these queries. We also design a cost metric for query route discovery in terms of energy efficiency of quorums used by nodes on a path. Simulation results indicate that our scheme can effectively conserve energy while keeping report latency short.

The rest of this paper is organized as follows. Preliminaries are given in Chapter 2. Chapter 3 presents our algorithms. Performance evaluations are given in Chapter 4. Finally, Chapter 5 concludes this paper.

Chapter 2

Preliminaries

2.1 Quorum-Based Power Saving Protocols

Power-saving protocols for wireless networks require to ensure nodes' wakeup patterns to have some degree of overlapping. A typical approach is to synchronize nodes' clocks. Recently, asynchronous protocols, which do not require nodes to synchronize their clocks, have been proposed based on the concept of *quorum*[4][11][14][22]. Fig. 2.1 shows an example of a grid quorum, which consists of $n_1 \times n_2$ elements arranged in an array and a node can pick any column plus any row of elements as its quorum. Each element can be regarded as a time slot. For each node, time axis is divided into time slots, which are partitioned into groups, each of $n_1 \times n_2$ slots. Each group is mapped to the array and whenever a quorum is encountered, the node has to stay awake. It has been shown that such a grid quorum can ensure at least some degree of overlapping between any two nodes even if their clocks are not synchronized. The concept has been applied to IEEE 802.11-based ad hoc network to support power saving mode [22]. In [4], it is shown that even if nodes use different arrays as their grid quorums, the overlapping property can still be guaranteed. Adaptive quorum-based MAC protocols have been proposed in [14][25]. Nodes can adjust their quorum size (i.e. array size) according to their traffic patterns.

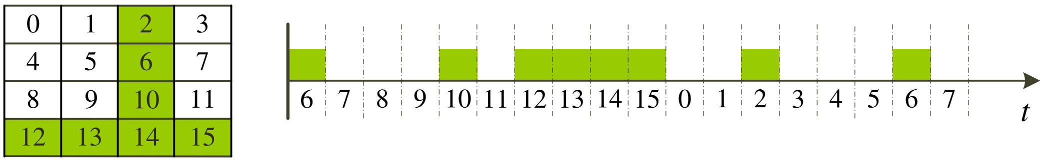

Figure 2.1: An example of grid quorum.

2.2 System Models

In this work, we will adopt the grid quorum system. A grid quorum Q is denoted by a 4-tuple (n_1, n_2, i, j), which means that an $n_1 \times n_2$ array is used and the i-th row and j-th column are picked as a quorum. Each sensor node v may have a set Q^v of quorums to define its wakeup pattern. More specifically, whenever a time slot is defined as an active slot by any grid quorum $Q \in Q^v$, then v has to stay awake in that slot. We assume that a time slot is long enough to complete at least a few packets exchange. A sensor that does not need to support any query can keep only one quorum in its quorum set to reduce its duty cycle. A node that needs to support multiple queries may need to keep multiple quorums in its quorum set. This will be clear in our protocol.

In this work, we consider continuous queries, which are defined as follows. each query q is denoted by a 5-tuple $(s_n, s_r, dur, int, len)$, where s_n is the sink, s_r is the data source, dur is the lifetime of q, int is the maximal interval that s_n expects to receive a report from s_r, and len is the report packet length according to what kind of data that sink want to retrieve.

Given a WSN with multiple continuous queries coming and leaving, our goal is to design a protocol to dynamically adjust the quorum sets of sensor nodes to meet the traffic demands of these queries while maintain low duty cycles of nodes.

Chapter 3

The Proposed Scheme

When a sink node s_n intends to subscribe continuous sensory data from a source node s_r, it will initiate a route discovery procedure by broadcasting a *route request (RREQ)* packet containing its query $q = (s_n, s_r, dur, int, len)$. An intermediate node, on receiving the RREQ, will process the packet and may further propagate the RREQ based on our energy metric. The source may receive several RREQs and will choose the route with the least cost and unicast a *route reply (RREP)* packet to the sink. The intermediate nodes on the path will adjust their quorum set properly. Then the source can transmit sensory data accordingly.

Each sensor node in the network has one default quorum $Q_d = (N, N, 1, 1)$ which indicates initially lower duty cycle in its quorum set to guarantee communication with other nodes. For simplicity sensor nodes select the first row and first column as quorum slots, and function according to their own quorum set. A node that needs to support any task for query will change the default quorum Q_d to a proper quorum Q' in compliance with the query requirements. Moreover, to facilitate the decision of all chosen quorums, we further define a set of m prime numbers, labeled by $\hat{P} = \{\hat{p}_1, \hat{p}_2, ..., \hat{p}_m\}$. The set $\hat{P}$ are used when deciding the default quorum Q_d and the proper quorum Q' to ensure that the grid quorum can be used to satisfy the interval requirement of the query. For example, the value of N could be chosen in any combination of elements of $\hat{P}$. $N = a_1 \times a_2 \times \cdots \times a_k$, where $\{a_1, a_2, ..., a_k\} \in \hat{P}$ and $a_1 \leq a_2 \leq \cdots \leq a_k$. Then the default duty cycle of nodes can be easily calculated as $\frac{N+N-1}{N^2}$.

In addition, a simple, inexpensive, *local synchronization* among nodes is proposed. This procedure described in Section 3.4 can adjust nodes' current quorum slot according to its neighbors and their statuses in order to make nodes have more chances to exchange packets with neighbors. In normal times it make the broadcast operation in quorum system more efficiently and in query times it can decrease query report latencies.

3.1 Route Request

Assume that the application layer a sink s_n generates a query $q_s = (s_n, s_r, dur, int)$. In this work, s_n determines a new int', where $int' \leq int$, by utilizing the prime numbers $\hat{P} = \{\hat{p}_1, \hat{p}_2, ..., \hat{p}_m\}$. The sink s_n first sets $int_{new} = int$ and then s_n performs a integer factorization algorithm to factorize int_{new}. If int_{new} is *not* factorable, s_n sets $int_{new} = int_{new} - 1$ and then tries to factorize the new int_{new} again. Otherwise, s_n can decide the int_{new}. Note that the above step is so designed to ensure that the grid quorum can be used to support the interval requirement of the query. We further restrict the new report interval can only be the combinations of these prime numbers. Because the quorums which the nodes on shared path used need to support multiple interval requirements in our algorithm. This design can increase the probability of sharing routing paths. Assume that $int_{new} = a_1 \times a_2 \times \cdots \times a_k$, where $\{a_1, a_2, ..., a_k\} \in \hat{P}$ and $a_1 \leq a_2 \leq \cdots \leq a_k$. Here, s_n temporally estimates a duty cycle of the relay nodes of q_s as

$$duty = \frac{a_1 \times a_2 \times \cdots \times a_{k-1} + a_k - 1}{a_1 \times a_2 \times \cdots \times a_k} \tag{3.1}$$

This *duty* is used to estimate the cost upper bound that all nodes of a routing path spend to be active, which will be discussed later. After finishing the above preparations, the sink s_n broadcasts an RREQ(*header*, $dur(q_s) = dur$, $int(q_s) = int_{new}$, $duty(q_s) =$ *duty*, $cost(q_s) = 0$, $shared(q_s) = 0$), where *header* contains dynamic source routing information, $cost(q_s)$ is the cost of q_s (initial to zero), and $shared(q_s)$ is a flag to indicate if the routing path has crossed with an existing route.

Assume that an intermediate node v receives an RREQ(*header*, $dur(q_i)$, $int(q_i)$, $duty(q_i)$, $cost(q_i)$, $shared(q_i)$) of query q_i. Node v first checks if it currently supports other query sessions. If so, assuming that there is one grid quorum $Q_k = (n_1^k, n_2^k, i^k, j^k)$ in $v's$ current quorum set Q^v, v checks the $shared(q_i)$ flag and do the following steps.

1. If $shared(q_i) = 0$, v firsts set a flag $shared' = 1$ and determines a variable $int' = int(q_i) - \min\{c\}$ such that $int' = m \times n_1^k \times n_2^k$, where c is a nonnegative integer and $m = 1, 1/2, 1/3, ..., 2, 3, ...$ Then v factorizes int' and computes the *duty* according to Eq. (3.1).

2. If $shared(q_i) = 1$ and $int' = m \times n_1^k \times n_2^k$, where $m = 1, 1/2, 1/3, ..., 2, 3, ...$, node v sets $int' = int(q_i)$ and *duty*$= duty(q_i)$.

3. Otherwise, v needs an extra grid quorum Q_k' into Q^v to support q_i, and then sets $int' = int(q_i)$ and *duty*$= duty(q_i)$.

Note that in Step 1, v checks if it can support this new query session by shortening the interval requirement recorded in the RREQ. In Step 2, $shared(q_i) = 1$ implies that the interval requirement of q_i has modified by previous nodes so v can easily support this query only when the modified requirement is the multiple of the size of Q_k or the size of Q_k is multiple of the modified requirement. However when there is an irreconcilable conflict in the interval requirement between queries, v can not still serve multiple queries using the original quorum. Thus step 3 ensure that the RREQ could be delivered reliably in high traffic load as nodes function according to more than one wakeup patterns. Absolutely, using more than one wakeup patterns which means an increase in duty cycle and more energy cost is the worst case we want to see.

Next, v estimates the cost when supporting q_i. v first calculates the packet cost as

$$packet\_cost = (C_{tx} - C_{active}) \times \frac{len}{data\_rate} \times \left\lceil \frac{dur(q_i)}{int'} \right\rceil, \tag{3.2}$$

where C_{active} and C_{tx} are the energy cost of sensors when in active and transmission mode, respectively. The packet cost represents the energy consumption of relaying report packets of this session. Then, v further calculates the additional cost, which is the energy consumption that v changes its quorum size, according to following steps.

1. If v currently does not supports any session,

$$additional\_cost = C_{active} \times (duty - \frac{2N-1}{N^2}) \times dur(q_i). \tag{3.3}$$

2. If v currently supports other sessions, there are two cases.

   (a) If q_i's finish time is later than the last finish time that v supports Q_k. v checks if $int' \geq n_1^k \times n_2^k$. If so,

$$additional\_cost = C_{active} \times (duty - \frac{2N-1}{N^2}) \times (dur(q_i) - t_v), \tag{3.4}$$

   where the t_v is the needed time for v to finish the last existing query session. Otherwise,

$$additional\_cost = C_{active} \times (duty - \frac{2N-1}{N^2}) \times (dur(q_i) - t_v) + C_{active} \times (duty - \frac{n_1^k + n_2^k - 1}{n_1^k \times n_2^k}) \times t_v. \tag{3.5}$$

   Fig. 3.1(a) shows an illustration of this cost calculation at node C. There is already a query session q_k, and the remaining lifetime is t_c. Eq. (3.4) is used when the new query session q_i could be fully supported by Q_k in the time period t_c and by Q'_k in the time period $dur(q_i) - t_c$, or node C should update Q_k to Q'_k in the time period t_c to support the two queries at the same time such as in Eq. (3.5).

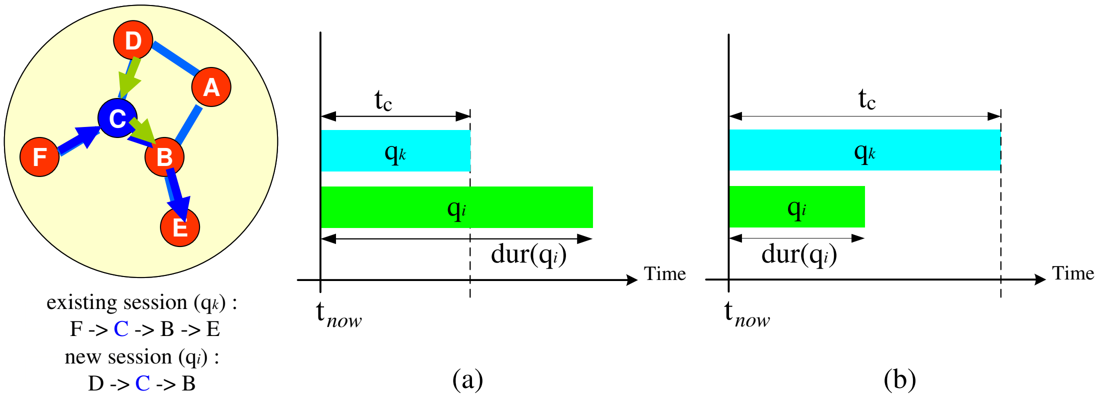

Figure 3.1: Two cases are handled in Cost Estimation when path overlaps.

(b) If q_i's finish time is *not* later than the last finish time of v's session. Again, v checks if $int \geq n_1^k \times n_2^k$. If so, *additional_cost* $= 0$. Otherwise,

$$\textit{additional\_cost} = C_{active} \times (duty - \frac{n_1^k + n_2^k - 1}{n_1^k \times n_2^k}) \times dur(q_i). \tag{3.6}$$

Fig. 3.1(b) shows an illustration of this cost calculation. Eq. (3.6) similarly shows that node C use Q'_k to support the two queries at the same time in the time period $int(q_i)$. C would need no *additional_cost* if it could support q_i during query lifetime without any change.

After obtaining *additional_cost*, v further needs to compute an compensative cost for relay nodes in RREQ when the variable $shared'$ is set.

$$\textit{comp\_cost} = C_{active} \times (\textit{duty} - duty(q_i)) \times dur(q_i) \tag{3.7}$$

When $shared'$ is *not* set, *comp_cost* $= 0$. The *comp_cost* represents the compensative energy consumption that those previous nodes which RREQ passed have to adjust their duty cycle to support this session when the interval requirement is modified. Node v can calculate a total cost of itself in two options.

- Option 1: $cost$ = *packet_cost* + *additional_cost*.

- Option 2: $cost = packet\_cost + additional\_cost - RE(v)$, where $RE(v)$ represents v's residual energy.

Note that this two options, node estimates the cost it should spend for the new query by Option 1 while estimates the residual energy after serving the new query by Option 2. Using Option 1, the energy consumption summation of the whole routing path is acquired. Thus we could choose the best route for the overall network to decrease energy wastes. On the other hand, it could achieve the load balance to prolong the network lifetime by our similar energy-aware routing when using Option 2. After estimating the cost, v performs the following steps to rebroadcast this RREQ.

1. v decides a *path_cost* as the following step.
   (a) If using Option 1, $path\_cost = cost(q_i) + comp\_cost \times hops + cost$, where *hops* means the number of traversed nodes record in *header*.
   (b) If using Option 2, $path\_cost = cost$.
2. If v has sent this RREQ, assuming the *path_cost* last time is $record\_cost(q_i)$, it performs the following steps.
   (a) When using Option 1 and the $path\_cost < record\_cost(q_i)$, v will rebroadcast this RREQ.
   (b) When using Option 2 and the $path\_cost < record\_cost(q_i) - comp\_cost$, v will rebroadcast this RREQ.
   (c) Otherwise, v discards this RREQ.
3. When $shared'$ is set, v broadcasts an RREQ(*updated_header*, $dur(q_i)$, int', *duty*, *path_cost*, 1).

4. When $shared'$ is not set, v broadcasts an RREQ(*updated_header*, $dur(q_i)$, int', *duty*, *path_cost*, $shared(q_i)$).

The RREQ will go on being rebroadcasted till the destination node s_r receives such an RREQ of q_i.

3.2 Route Reply

When s_r receives a new RREQ, it waits a time period for other RREQs with multi-path. After timeout, the s_r keeps the RREQ with the least path cost. Considering the $int(q_i)$ field in RREQ, the s_r distinctly decides a quorum $Q_p = (n_1^p, n_2^p, i^p, j^p)$ by the following steps, where $n_1^p \times n_2^p = int(q_i)$ which indicates a lowest duty cycle to serve q_i and i^p, j^p are randomly selected row and column to wake up and transmit or receive packets.

1. s_r determines a variable $b_1 = \lfloor \sqrt{int(q_i)} \rfloor - \min\{c\}$ such that $b_1 = a_1 \times a_2 \times \cdots \times a_k$, where c is a nonnegative integer, $\{a_1, a_2, ..., a_k\} \in \hat{P}$ and $a_1 \leq a_2 \leq \cdots \leq a_k$.

2. s_r gets another variable b_2 such that $int(q_i) = b_1 \times b_2$. If $b_1 \geq b_2$, s_r sets $(n_1^p, n_2^p) = (b_1, b_2)$. Otherwise, s_r sets $(n_1^p, n_2^p) = (b_2, b_1)$.

Note that in the above steps, s_r takes a factorization, which is composed of two values that are closed to the square of $int(q_i)$. By this design, s_r can find a quorum, which induces a smaller duty cycle. For example, assuming that $int(q_i) = 24$ and $\hat{P} = \{2, 3, 5\}$, we can have a factorization $int(q_i) = 2 \times 2 \times 2 \times 3$. When setting $\{n_1 \times n_2\}$ to $\{12 \times 2\}$, $\{8 \times 3\}$, and $\{6 \times 4\}$, the duty cycle will be 54%, 42%, and 38%, respectively. In this case, s_r sets $\{n_1 \times n_2\} = \{6 \times 4\}$. Also note that step 2 not only produces continuous quorum slots for sensors but tends to alleviate channel contention and packet collision while reporting query results. It is obviously if the sensors that located on a routing path start their continuous slots about the same timing, the report data can quickly arrive the sink with high probability.

Table 3.1: Query Table

Query ID	Query	Upstream Node	Downstream Node	Quorum
12	$(31, 99, 2000, 40, 100)$	55	129	$(8, 5, 1, 1)$
7	$(101, 29, 1000, 20, 100)$	63	129	$(5, 4, 3, 3)$
$\ldots$	$\ldots$	$\ldots$		

Then, the s_r replies a route reply packet RREP(*header*, q_i, Q_p). This RREP sequentially traverses the routing path recorded in *header*. Assume the intermediate node v, which is recorded in the RREP, receives this packet. v keeps the information of q_i and Q_p, then relays this RREP to the next hop node in the *header*. Every node will maintain the information and corresponding quorums of all queries passing itself, such as Table 3.1, in order to handle the route remove procedure in Section 3.3. If there is only default quorum Q_d in Q^v, Q_p directly replaces Q_d. When v is going to serve multiple queries, assuming there is already a quorum Q_q in Q^v, the quorum group size is denoted as $|Q_q|$, v checks if one of the two quorums Q_p and Q_q could be merged into the other so that $Q^v = \{Q_q\}$ if $|Q_p| = c|Q_q|$ or $Q^v = \{Q_p\}$ if $|Q_q| = c|Q_p|$, where c is a natural number. Suppose they could not be merged, the quorum set becomes $Q^v = \{Q_p, Q_q\}$. Put it differently, v should spend more energy to perform a multi-quorum sleeping schedule. This interprets our design idea that the relationship between quorum and query could be one-one or one-many relation.

When the sink receives the RREP, it sends an Announce packet to start the continuous query q_i. The relay nodes of this query will start their new wake-up patterns following their own quorum sets when receives the Announce packet.

3.3 Route Remove

As sensor node maintain query information served, it should update its query table once one of these continuous queries is going to expire. The data source will sends an RREM packet to

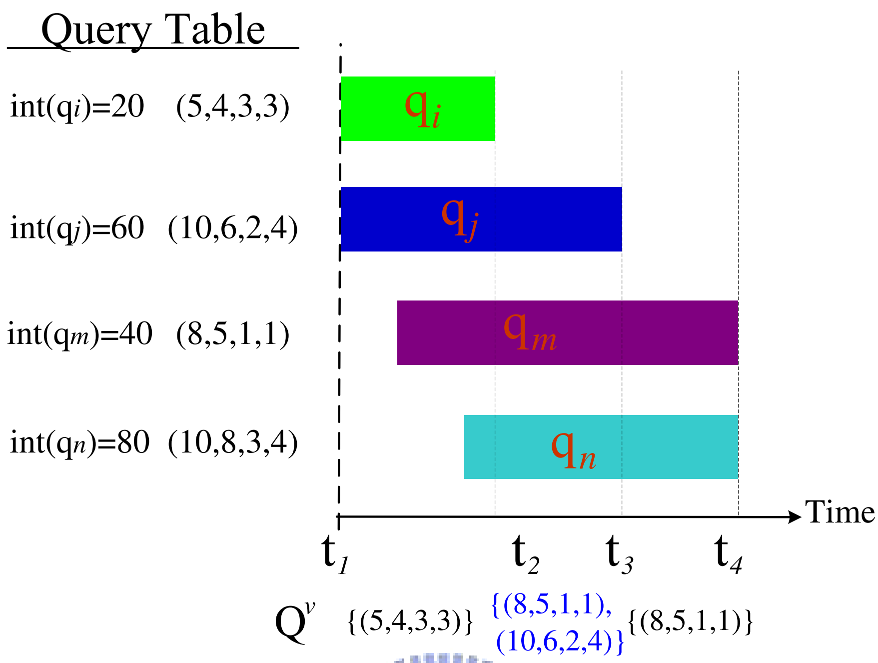

Figure 3.2: The transition of quorum group Q^v in a sensor node v.

terminate the session. An intermediate node v receiving the RREM removes the corresponding entry in query table and adjusts the corresponding element in its quorum set to serve the other continuous queries that are still executing. Fig. 3.2 shows an example of the process of route remove. v serves two queries q_i and q_j with $Q^v = \{Q_i\}$ ($|Q_i| = 20$) in the time t_1. Q^v does not change after serving q_m and q_n in sequence as all the quorums(Q_j, Q_m, Q_n) could be merged into Q_i. v receives an RREM packet in the time t_2 when q_i is going to terminate, it finds a new quorum set Q^v to satisfy the remaining three queries, i.e. $Q^v = \{Q_j, Q_m\}$. By the same token, q_j terminates in the time t_3, $Q^v = \{Q_m\}$ is the new quorum set. If the only query which v served terminates, $v's$ quorum set resets to $Q^v = \{Q_d\}$ as the condition in the time t_4. Similarly v then starts new wake-up pattern following its own quorum set every time when its quorum set changes.

3.4 Local synchronization

We propose a *local synchronization* procedure to let nodes can meet their neighbors in continuous slots. Nodes periodically exchange HELLO(*ID*, *HasASession*, *CurSlotNum*) packets with neighbors. This HELLO only broadcast in quorum slots. Assume a node v receives multiple HELLOs from its neighbors. v performs the following steps.

1. If v currently does not supports any session, v finds a node u that satisfies the following conditions: 1) u neither supports any session, i.e. *HasASession*=0 and 2) u's ID is smaller than v's ID. If there are multiple candidates, v sets the node with the smallest ID as u. Then go to step 3.

2. If v currently supports one or more sessions, v could find a partial neighbor set N^v consisting of all $v's$ upstream and downstream nodes whose node ID is less than v by its query table. If N^v is a null set, v will ignore this *local synchronization* procedure. Each node in N^v has a priority based on quorum size and node ID. The node with the smallest quorum size and the smallest ID has the highest priority. Then v appoints one node which belongs to N^v as u in the order of priority. Then go to step 3.

3. Assume the current quorum slot number of v and u are cur_v and cur_u, respectively. v modifies the number of its next slot as $cur_u + 1$ instead of $cur_v + 1$. And v performs a wake-up pattern from its quorum set Q^v as usual.

As mentioned earlier, a data packet should be forwarded multiple hops within successive slots to decrease the total transmission delay. Clock drift is not a rare situation in multi-hop WSNs, as distributed clock synchronization is a nontrivial issue. Especially in contention-based network, the random backoff mechanism may result in temporary link failure, so that the upstream node is forced to queue data packet in its buffer and waits next common active slot for retransmission. In other words, if neighbor discovery takes most of time, it not only increases the end-to-end latency certainly but burdens battery-limited sensor with much idle listening. Put the successive time slots of two nodes close prevents the occurrence of link failure effectively. Therefore the waiting time before one node could communicate with its downstream node is eliminated precisely this way.

Chapter 4

Performance Evaluation

In order to justify our query processing system, we compare our system against SMAC[24], PMAC[25], and TAP[9]. As there are two options that could be used in our algorithm, we use $Quorum_1$ to represent the one using Option 1 and $Quorum_2$ to represent the other. Unlike TAP[9] and our system, SMAC[24] and PMAC[25] are not cross-layer designs. Hence, based on SMAC[24] and PMAC[25], we simulate the AODV protocol to route query data packets. The simulated network is 10 topologies which are randomly conducted in a $400 \times 400\ m^2$ square area with more than 200 randomly distributed stationary sensors. Each sensor node has transmission range and carrier sensing range 50 and 100 meters respectively.

In our simulations, we will generate six sink-source query pairs in random networks with random query requirements in sequence. There is a mean interval of 1000 seconds between two queries. The whole simulation time will continue to 7200 seconds. Each query data packet is assumed 50 bytes in size. In order to verify the energy efficiency of our system, we set the same duty cycle for all simulated protocols (5% as the default setting). The power consumption rates of the wireless module are set 50, 50, 45 and 5 mW in transmit, receive, idle and sleep modes respectively. Table 4.1 lists the key parameters we used in our simulations. Meanwhile, in order to show the benefit brought up by traffic awareness, we also compared the routing performance using the end-to-end delay as the metric. The simulation results are depicted in the following figures, which will be discussed in detail. Intermediate relaying

Table 4.1: Network parameters

Sensor Initial Power	100 J
Tx Power	50 mW
Rx Power	50 mW
Idle Power	45 mW
Sleep Power	5 mW
Tx Range	50 m
Carrier Sensing Range	100 m

nodes do not aggregate or compress data. We also assume that the application data processing at any node introduces little extra delivery delays.

4.1 Energy Consumption Evaluation

As mentioned procedure in the previous section, Fig. 4.1 represents one simulated routing result. We can see clearly that our system brings in an effect of path-sharing. It did make use of the idle time slots to relay other query data streaming and decrease the energy wasted in vain. After simulation time terminates, the percentage of the nodes' residual energy display in the Fig. 4.2.

Fig. 4.3 also shows the comparison of each protocol in the power consumption of each node. Because PMAC[25] uses a sleep strategy under an unfixed schedule, it can save more energy than other system. However PMAC[25] will incur extreme end-to-end query report latency which will be corroborated in Fig. 4.4. For multi-hop delivery of a series of report packets, several sensor nodes in our system could utilize the idle slots on the existent paths to serve different queries. The path-sharing utilizing idle slots can not only let most nodes still operate in default lower duty cycle, but also increase the energy efficiency of the entire network. Also, in Fig. 4.3, we can see one trend: as the density of nodes increases, the average hop count of routing path becomes smaller. Hence, the chance of utilizing the idle slots on the

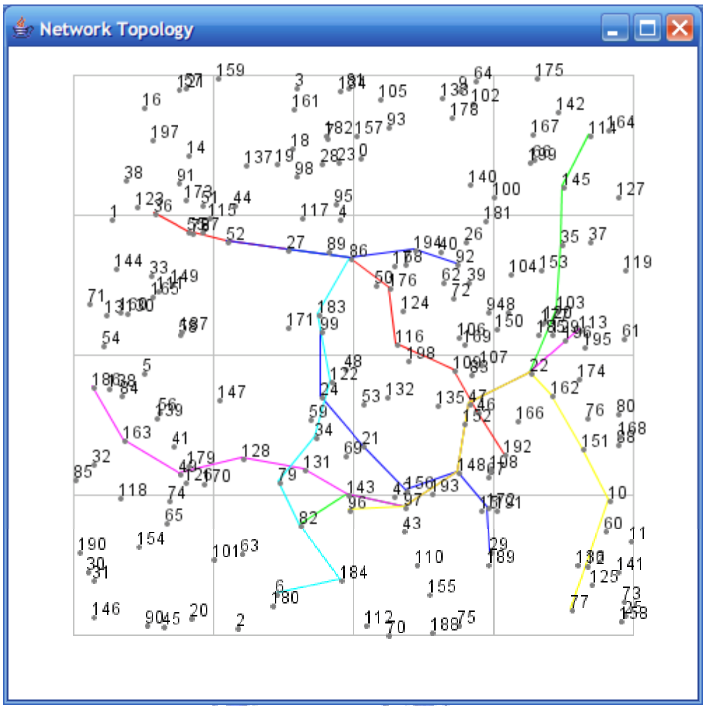

Figure 4.1: One random simulation scenario.

existent query paths to cut down energy cost will reduce.

Fig. 4.5 shows the average residual power of all the sensors. Average residual power is determined as the sum of all nodes' residual energy divided by the number of nodes. Nodes in PMAC[25] move into longer sleep mode rapidly when there is no traffic in the network, thus having better power efficiency. As the traffic comes, node changes to a schedule without any sleep time and waits for the next hop to be active. Such a situation is avoidable in our system.

4.2 Latency Evaluation

In this subsection, we evaluate the performance of average end-to-end delivery latency. TAP[9] is a routing scheme utilizing the traffic awareness. The routing path constructed by it is always the shortest one in existent network topology, thus it has the lowest end-to-end latency. Hence, Fig. 4.4 unfolds the results of our latency evaluation. It is showed in a log scale, that is to say,

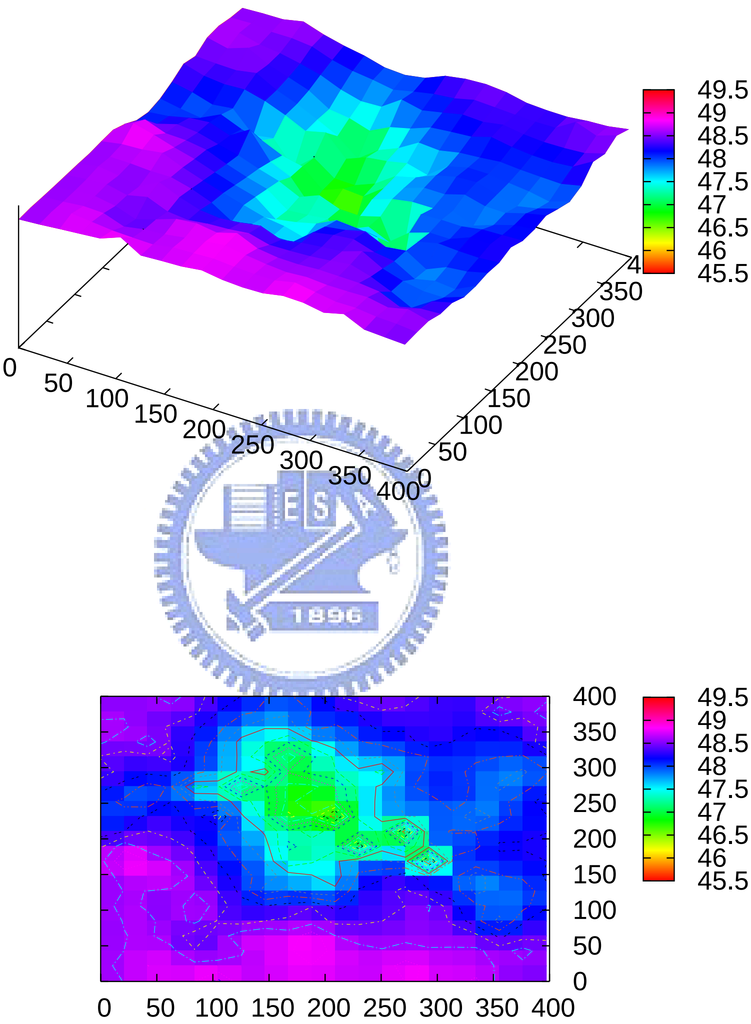

Figure 4.2: The percentage of the residual energy of nodes after six query requests occur.

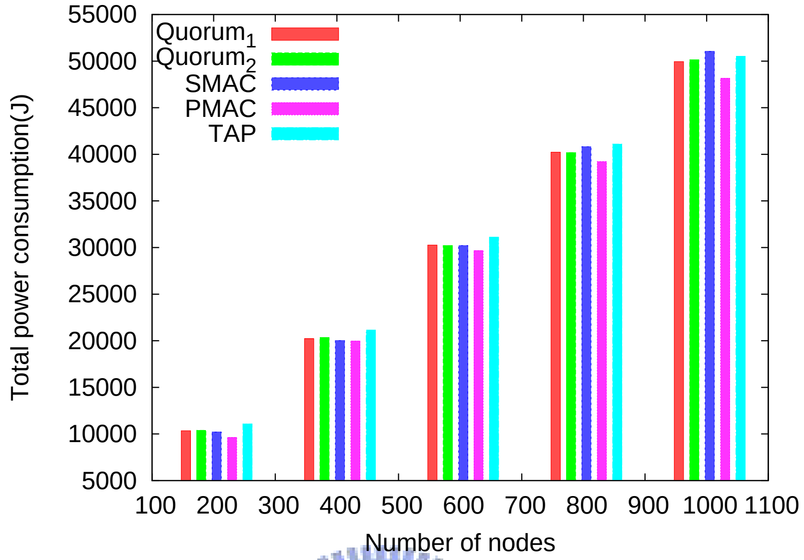

Figure 4.3: Total power consumption of network

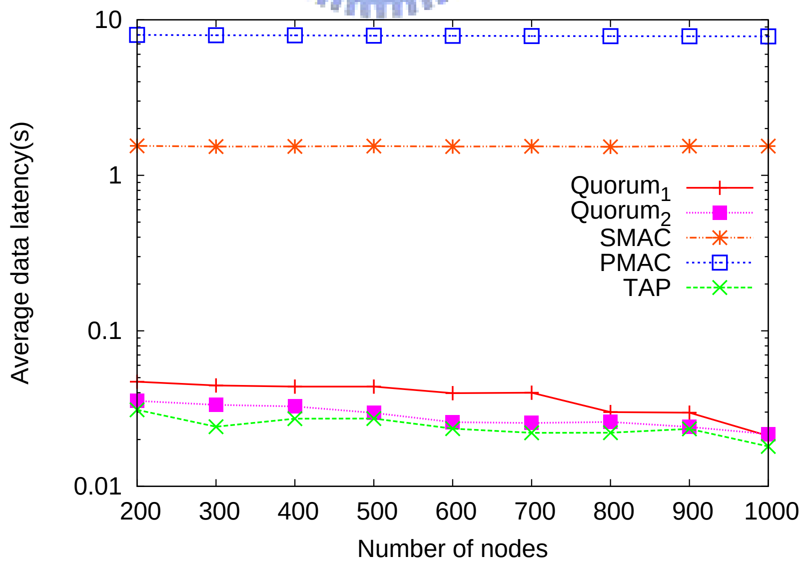

Figure 4.4: Data delivery latency

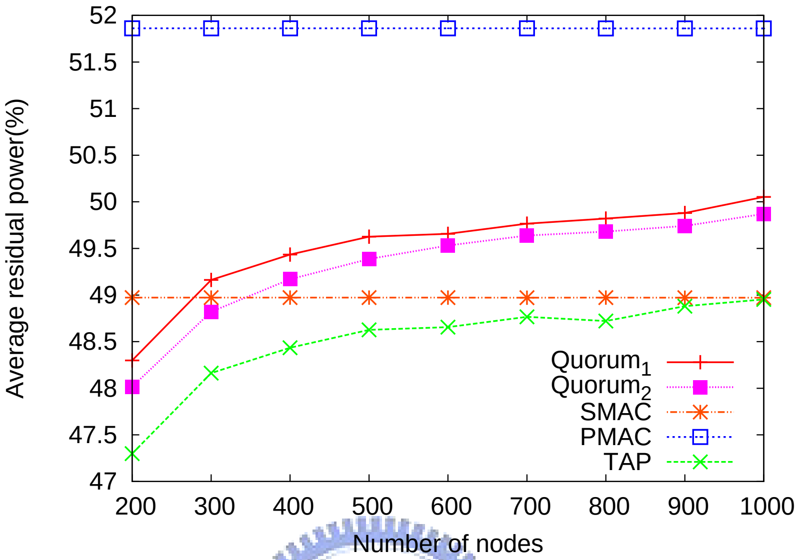

Figure 4.5: Average residual power of sensors

Table 4.2: The minimum value of sensors' residual power(%)

Size	$Quorum_1$	$Quorum_2$	$SMAC$	$PMAC$	TAP
200	23.97	33.49	47.96	50.75	0
300	36.85	32.93	47.97	50.75	0
400	25.65	32.93	47.97	50.71	0
500	20.61	37.97	47.97	50.75	0
600	27.37	34.29	47.96	50.71	0
700	29.01	33.49	47.96	50.72	0
800	29.33	36.61	47.97	50.72	0
900	26.13	32.93	47.96	50.72	0
1000	29.91	37.49	47.96	50.75	0

there is little difference in the metric of end-to-end delivery latency between TAP[9] and our quorum cross-layer scheme.

And in Table 4.2, it shows the minimum values for a single sensor's residual power. TAP[9] might cause the network partition by keeping some node active long time to maintain local connectivity. When using Option 2 in our system, it use the residual energy of nodes on path to evaluate the path cost. However, it does achieve more load balance between nodes up to 10% than using Option 1. Therefore, our system is not only more energy efficient than TAP[9], but has shorter end-to-end delays similar to TAP[9].

Chapter 5

Conclusions

We have developed a query processing system supporting multiple continuous queries simultaneously in the wireless sensor network. Our design emphasizes on routing paths overlapping and quick query results without extra delay. We modify the original DSR routing scheme adding a proposed cost metric for operating in coordination with the quorum concept to satisfy requirements of multiple query simultaneously. Simulation results are reported to verify the correctness and performance of the proposed scheme. It did decreases more energy consumption than other schemes without integrated routing algorithm and incur lower latency at the same time. Future research directions including extending our results to a high-traffic environments or applying a moving model to model our observers(sinks).

Bibliography

[1] J. Bachrach, R. Nagpal, M. Salib, and H. Shrobe. Experimental results and theoretical analysis of a self-organizing global coordinate system for ad hoc sensor networks. *Telecommunications Systems Journal*, 26(2-4):213–234, 2004.

[2] S.-J. Baek, G. de Veciana, and X. Su. Minimizing energy consumption in large-scale sensor networks through distributed data compression and hierarchical aggregation. *IEEE Journal on Selected Areas in Communications*, 22(6):1130–1140, 2004.

[3] D. Braginsky and D. Estrin. Rumor routing algorithm for sensor networks. In *Proc. of ACM Int'l Workshop on Wireless Sensor Networks and Applications (WSNA)*, 2002.

[4] C.-M. Chao, J.-P. Sheu, and I.-C. Chou. An adaptive quorum-based energy conserving protocol for ieee 802.11 ad hoc networks. *IEEE Trans. Mobile Computing*, 5(5):560–570, 2006.

[5] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris. Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks. In *Proc. of ACM Int'l Conference on Mobile Computing and Networking (MobiCom)*, 2001.

[6] S. Du, A. K. Saha, and D. B. Johnson. Rmac: A routing-enhanced duty-cycle MAC protocol for wireless sensor networks. In *Proc. of IEEE INFOCOM*, 2007.

[7] D. Ganesan, B. Greenstein, D. Perelyubskiy, D. Estrin, and J. Heidemann. An evaluation of multiresolution storage for sensor networks. In *Proc. of ACM Int'l Conference on Embedded Networked Sensor Systems (SenSys)*, Los Angeles, USA, 2003.

[8] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-efficient communication protocols for wireless microsensor networks. In *Proc. of Hawaii Int'l Conference on Systems Science (HICSS)*, 2000.

[9] P. Hu, P.-L. Hong, J.-S. Li, and Z.-Q. Qin. Tap: Traffic-aware topology control in on-demand ad hoc networks. *Computer Networks, Elsevier Science Pub.*, 29(18):3877–3885, 2006.

[10] C.-F. Huang, Y.-C. Tseng, and L.-C. Lo. The coverage problem in three-dimensional wireless sensor networks. In *Proc. of IEEE Global Telecommunications Conference (Globecom)*, 2004.

[11] J.-R. Jiang, Y.-C. Tseng, C.-S. Hsu, and T.-H. Lai. Quorum-based asynchronous power-saving protocols for ieee 802.11 ad hoc networks. *ACM/Kluwer Mobile Networks and Applications*, 10(1/2):169–181, 2005.

[12] M. Kochhal, L. Schwiebert, and S. Gupta. Role-based hierarchical self organization for wireless ad hoc sensor networks. In *Proc. of ACM Int'l Workshop on Wireless Sensor Networks and Applications (WSNA)*, 2003.

[13] Q. Li, M. DeRosa, and D. Rus. Distributed algorithm for guiding navigation across a sensor network. In *Proc. of ACM Int'l Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, Maryland, USA, 2003.

[14] W.-H. Liao, H.-H. Wang, and W.-C. Wu. An adaptive MAC protocol for wireless sensor networks. In *Proc. of IEEE Int'l Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, 2007.

[15] S. Meguerdichian, F. Koushanfar, M. Potkonjak, and M. B. Srivastava. Coverage problems in wireless ad-hoc sensor networks. In *Proc. of IEEE INFOCOM*, 2001.

[16] D. Niculescu and B. Nath. DV based positioning in ad hoc networks. *Telecommunications Systems Journal*, 22(1-4):267–280, 2003.

[17] M.-S. Pan, C.-H. Tsai, and Y.-C. Tseng. Emergency guiding and monitoring applications in indoor 3d environments by wireless sensor networks. *Int'l Journal of Sensor Networks (IJSnet)*, 1(1/2):2–10, 2006.

[18] M.-S. Pan, L.-W. Yeh, Y.-A. Chen, Y.-H. Lin, and Y.-C. Tseng. Design and implementation of a wsn-based intelligent light control system. In *Proc. of IEEE Int'l Workshop on Wireless Ad hoc and Sensor Networks (WWASN)*, 2008.

[19] H. Park, M. B. Srivastava, and J. Burke. Design and implementation of a wireless sensor network for intelligent light control. In *Proc. of ACM/IEEE Int'l Conference on Information Processing in Sensor Networks (IPSN)*, 2007.

[20] K. Sohrabi, J. Gao, V. Ailawadhi, and G. J. Pottie. Protocols for self-organization of a wireless sensor network. *IEEE Personal Communications*, 7(5):16–27, October 2000.

[21] R. Szewczyk, A. Mainwaring, J. Polastre, J. Anderson, and D. Culler. An analysis of a large scale habitat monitoring application. In *Proc. of ACM Int'l Conference on Embedded Networked Sensor Systems (SenSys)*, 2004.

[22] Y.-C. Tseng, C.-S. Hsu, and T.-Y. Hsieh. Power-saving protocols for ieee 802.11-based multi-hop ad hoc networks. *Computer Networks, Elsevier Science Pub.*, 43(3):317–337, 2003.

[23] Y. Xu, J. Heidemann, and D. Estrin. Geography-informed energy conservation for ad hoc routing. In *Proc. of ACM Int'l Conference on Mobile Computing and Networking (MobiCom)*, 2001.

[24] W. Ye, J. Heidemann, and D. Estrin. An energy-efficient MAC protocol for wireless sensor networks. In *Proc. of IEEE INFOCOM*, 2002.

[25] T. Zheng, S. Radhakrishnan, and V. Sarangan. Pmac: An adaptive energy-efficient MAC protocol for wireless sensor networks. In *Proc. of IEEE Int'l Parallel and Distributed Processing Symposium (IPDPS)*, 2005.

Curriculum Vita

Tsu-Wen Hsu (tsuwen@cs.nctu.edu.tw) received his B.S. degree in Computer Science from National Chiao-Tung University, Taiwan, in 2006. His research interests include wireless networks, sensor networks and resource management.