Wireless Pers Commun (2009) 51:411–426
DOI 10.1007/s11277-009-9760-x

Cross-Layer, Energy-Efficient Design for Supporting Continuous Queries in Wireless Sensor Networks: A Quorum-Based Approach

Chia-Hung Tsai · Tsu-Wen Hsu · Meng-Shiuan Pan · Yu-Chee Tseng

Abstract Power saving and query processing are two major concerns in a wireless sensor network. Each of these two issues has been intensively studied separately in the literature. In this work, we are interested in linking the *asynchronous power-saving protocol* and the *continuous query-processing problem* together. A cross-layer solution is proposed. On the MAC layer, we propose to use the *grid-quorum system* (Tseng et al., Computer Networks, 43(3):317–337, 2003) to serve as the underlying power-saving framework. On the network layer, we propose to find query paths based on the power cost incurred by grid quorums used by nodes along a path. We show how these two layers interwork with each other to support continuous queries in an energy-efficient way.

Keywords Power saving · Protocol design · Query processing · Routing · Wireless sensor network

1 Introduction

The rapid progress of wireless communication and MEMS technology have 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. Recently, a lot of research works have been dedicated to WSNs, such as routing [6,9], self-organization [12,23], deployment [8,16,28],

C.-H. Tsai (✉) · T.-W. Hsu · M.-S. Pan · Y.-C. Tseng
Department of Computer Science, National Chiao Tung University, 1001 Ta Hsueh Road, Hsinchu 30010, Taiwan, ROC
e-mail: chiahung@cs.nctu.edu.tw

T.-W. Hsu
e-mail: tsuwen@cs.nctu.edu.tw

M.-S. Pan
e-mail: mspan@cs.nctu.edu.tw

Y.-C. Tseng
e-mail: yctseng@cs.nctu.edu.tw

and localization [5,20]. Applications of WSNs include emergency guiding [13,26], light control [17,18], and environment monitoring [24].

Power saving and *query processing* are two main issues in WSNs. Many power-saving MAC protocols have been proposed. In SMAC [30], nodes periodically switch to sleep mode. In PMAC [31], sensors are allowed to adaptively determine their sleep schedules by considering neighbors' traffic patterns. In RMAC [3], sensor nodes periodically wake up and use their active periods to establish routing paths. Nodes not located on any routing path can go to sleep; otherwise, they have to remain active. GAF [29] divides the network area into square grids. Although sensors can switch between sleep mode and active mode periodically, GAF guarantees that at least one node per grid remains active to exchange packets with neighboring grids. Span [2] 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 [2] and [29] may have some redundant sensors to stay active. TAP [7] considers traffic flows and identifies redundant nodes that can go to sleep when establishing routing paths. Most of these schemes require nodes to be synchronized in time, which are costly. Recently, some power-saving protocols have been proposed without requiring time synchronization [1,11,14,24,25].

On the other hand, query processing in WSNs has also attracted a lot of attention. Directed diffusion [10] achieves energy efficiency by selecting empirically good paths and by caching and processing data inside the network. In [19], data-centric storage is proposed by adopting geographic hashing to offer high data availability and load distribution. TAG [15] is a tiny data service that can significantly reduce bandwidth consumption. A semistructure approach which uses multiple shortest-path trees is proposed in [4] to support scalable data aggregation. A lot of works [21,22] utilize the spatio-temporal correlations of sensing data to achieve energy efficiency. A generic two-tier storage strategy for answering precision-constrained approximate queries is proposed in [27]. Although most of these query-processing works focus on achieving energy efficiency, they all do not specifically address the underlying wake-up/sleep schedules of sensor nodes.

In this work, we are interested in applying the *quorum-based* power-saving protocols [1,11,14,25], which have the advantage of not relying on any time synchronization among sensor nodes, to the *continuous query-processing problem*. A continuous query involves sending periodical reports from a source to a sink and is commonly seen in WSNs. More specifically, we will adopt the *grid-quorum* system [25] to derive the wake-up/sleep schedules of sensor nodes. Multiple query paths may coexist, each with its preferred grid quorum. We will show how these paths (and thus grid quorums) interact with each other to meet each query's bandwidth requirement in an energy-efficient way. Although global clock synchronization is not necessary, we will suggest to employ an optional *local slot synchronization* to improve nodes' energy efficiency. Compared to existing works, this paper contributes in proposing a cross-layer approach to integrate the grid-quorum system with continuous queries. Simulation results are presented to evaluate our results.

The rest of this paper is organized as follows. Section 2 presents our cross-layer system architecture. The detail MAC layer (quorum layer) and network layer (query-processing layer) are presented in Sects. 3 and 4, respectively. Section 5 contains our simulation results. Finally, Sect. 6 concludes this paper.

2 System Architecture

We are given a WSN for supporting continuous queries. A *continuous query* is a unicast with sensing data being periodically delivered from a source node to a sink node. A continuous

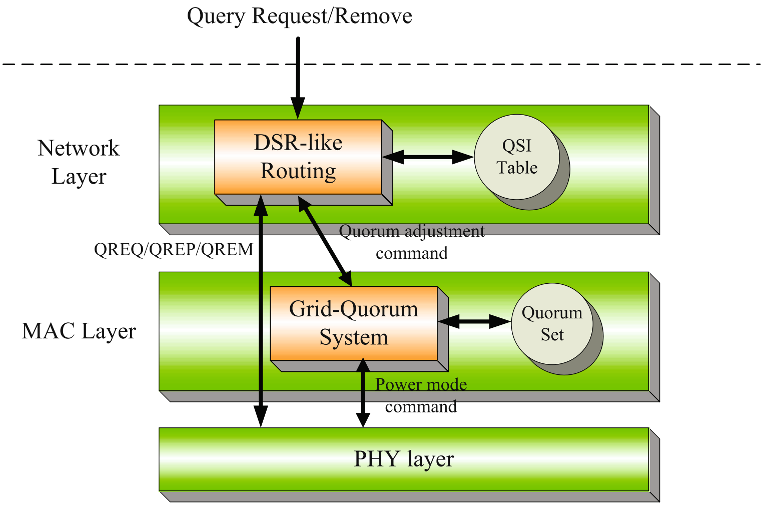

Fig. 1 The proposed 2-layer architecture

query, or simply query, is denoted by a 5-tuple (s_n, s_r, t, p, len), where s_n is the sink node, s_r is the source node, t is the lifetime of the query, p is the period that s_r will generate reports, and len is the expected packet length per report. Multiple queries may coexist in the network. We use the grid-quorum system [25] as the underlying MAC layer to support power management and develop a routing layer on the top of the quorum system to determine its parameters. The goal is to support continuous queries in an energy-efficient manner.

We propose a 2-layer architecture as shown in Fig. 1. When a continuous query arrives at the network layer, the sink will broadcast a query request (*QREQ*) packet to find a reporting path to the source. Such QREQ packets will be flooded around the network. To reply, the source will unicast a query reply (*QREP*) packet to the sink. To save sensor nodes' energy, a cost function is designed at the network layer to select query paths and to dynamically choose/adjust the quorum system's parameters. Then, the MAC layer will give power mode commands to the underlying layer. Note that when there are multiple queries, our cross-layer approach will try to increase the overlapping among nodes' quorums to reduce the energy costs to support these queries. After a query expires, a query remove (*QREM*) packet will be sent along its query path.

Each node will maintain a *Query Session Information (QSI)* table to keep track of the query paths that currently pass it and the quorums to support these paths. Table 1 shows the structure of the QSI table. Gird quorums in this table will together form the quorum set of the node. The detail MAC-layer and network-layer operations will be discussed in Sects. 3 and 4, respectively.

Table 1 An example of the QSI table

Query	Up_Node	Down_Node	Quorum	Additional_Quorum
(31, 99, 2000, 40, 100)	55	129	$(8, 5, \{1\}, \{1\})$	ϕ
(101, 29, 1000, 20, 100)	63	129	$(5, 4, \{3\}, \{3\})$	ϕ

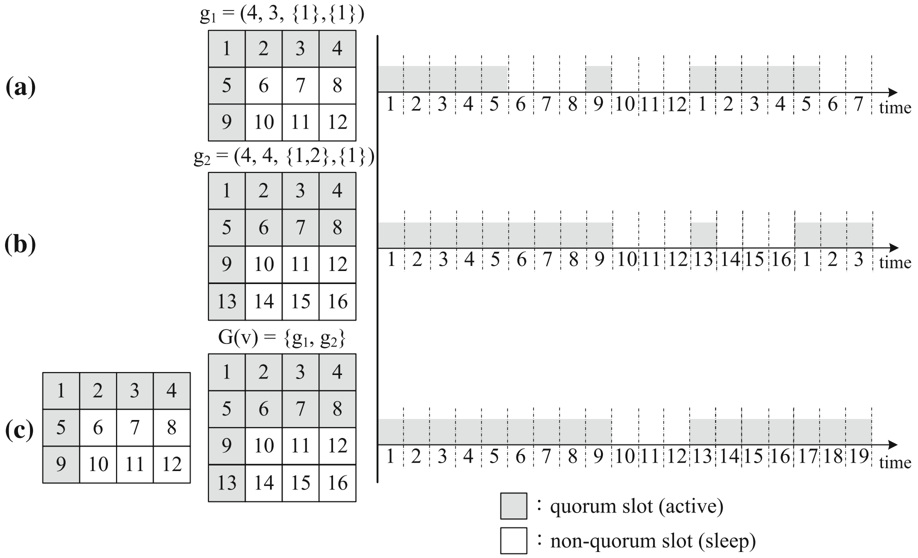

Fig. 2 An example of a quorum set

3 Quorum Layer

3.1 Grid Quorum System

Power-saving protocols for wireless networks need to ensure that nodes' wake-up patterns will overlap with their neighbors' patterns for communication opportunity. It is pointed out in [25] that two major challenges that one would encounter when designing a power-saving protocol are: *clock synchronization* and *neighbor discovery*. Therefore, many solutions try to enforce nodes to synchronize their clocks. However, time synchronization in a large-scale distributed environment is very costly. An alternative is to develop asynchronous power-saving protocols. The *quorum-based* protocols [1,11,14,25] are such solutions. Basically, they require nodes to wake up and sleep based on some pre-configured rules, but nodes do not need to synchronize their clocks. Several kinds of quorums have been proposed, such as tree quorums and grid quorums.

In this work, we will adopt the grid-quorum system [25] as our power-saving mechanism. Figure 2a shows a grid-quorum example. Each node's time axis is divided into repetitive $n_1 \times n_2$ time slots, which are called a *group*. In each group, its slots are arranged as an $n_1 \times n_2$ array in a row-major manner. From the array, the node can arbitrarily pick one column and one row of slots as its wake-up slots, or called *quorum slots*. Each node must stay awake in quorum slots, and can go to sleep in the remaining $n_1 \times n_2 - n_1 - n_2 + 1$ slots. Note that nodes' clocks do not need to be synchronized.

The concept has been applied to IEEE 802.11-based ad hoc networks in [25] by enforcing all nodes to take the same values of n_1 and n_2. In [1], it is further shown that even if two nodes use different n_1 and n_2, transmission opportunity (i.e., overlapping of wake-up patterns) between them is still guaranteed.

3.2 Quorum Set for Continuous Queries

In this work, we are interested in applying the grid-quorum system to support continuous queries in a WSN. The wake-up/sleep schedule of a node will be determined by one or multiple grid quorums, which we call *quorum set*. The quorum set of a node v is denoted by $G(v)$. Each grid quorum is denoted by a 4-tuple $g = (n_1, n_2, R, C)$, where n_1 and n_2 are the numbers of rows and columns, respectively, of the grid array, R is a set of rows, and C is a set of columns. Note that this is an extension of the original definition in [25] since all entries falling in rows of R or columns of C are quorum slots. We define the duty cycle of a grid quorum $g = (n_1, n_2, R, C)$ by

$$dty(g) = \frac{|R| \times n_1 + |C| \times n_2 - |R| \times |C|}{n_1 \times n_2}. \tag{1}$$

For example, Fig. 2a shows a grid quorum g_1 following the original definition of [25] (it contains only one row and one column of quorum slots). In Fig. 2b, $g_2 = (4, 4, \{1, 2\}, \{1\})$ is an extended grid quorum, which contains two rows and one column of quorum slots. Figure 2c shows a quorum set $G(v) = \{g_1, g_2\}$, in which case, v will run both quorums g_1 and g_2 simultaneously by “OR” the quorum slots of both g_1 and g_2. That is, whenever any of the grid quorums in $G(v)$ indicates that a slot is a quorum slot, v will enter the active mode. So Fig. 2c is the “OR” of the two sequences in Fig. 2a, b.

4 Query-Processing Layer

In our system, when a node does not support any continuous query, its quorum set will contain only one default grid quorum g_{def} with minimum duty cycle. As more and more continuous queries (query paths) pass the node, its quorum set will contain more grid quorums. The default quorum is defined as $g_{def} = (n_{max}, n_{max}, \{rnd\}, \{rnd\})$, where n_{max} is a large number and rnd is a random integer between 1 and n_{max}. A DSR-like routing protocol will be applied. To select a routing path, an energy cost function will be defined to evaluate the quality of a query path. Basically, a new path will try to increase its overlapping of quorum slots with existing paths’ quorum slots while maintain sufficient communication capacity.

Section 4.1 presents the query-requesting process, followed by the query-replying and the query-removing processes in Sects. 4.2 and 4.3. Finally, in Sect. 4.4, a lightweight local slot synchronization is proposed to increase energy efficiency.

4.1 Query-Requesting Process

This part contains three modules, *quorum preparing*, *QREQ initiating and processing*, and *QREQ rebroadcasting*, as explained below.

4.1.1 Quorum Preparing

When a sink node s_n has a query $y = (s_n, s_r, t, p, len)$ to a source node s_r, it will compute a grid quorum g_{ini} to support the query y as follows. Here we assume that from past history, the length len per report is already known.

(1) Compute a pair (n_1, n_2) such that $n_1 \times n_2 \approx p$ and n_1 is as close to n_2 as possible.

(2) Construct a grid quorum $g_{ini} = (n_1, n_2, R, C)$, where R/C contains a random row/column.

(3) Then, we check whether $dty(g_{ini}) \geq \frac{len}{r} \times \frac{1}{p}$ holds, where r is the transmission rate of a node. If so, we will adopt g_{ini} as the grid quorum to serve the query y. Otherwise, we will continuously add rows or columns to R or C to increase the duty cycle value $dty(g_{ini})$, until $dty(g_{ini}) \geq \frac{len}{r} \times \frac{1}{p}$ holds.

Note that g_{ini} is only considered as a candidate to support y; it may or may not be actually used on the query path between s_r and s_n. This will become clear later.

4.1.2 QREQ Initiating and Processing

There are two cases involving in producing a QREQ packet: (i) a node initiates a new query and (ii) a node receives a QREQ and rebroadcasts it. Below, we will only consider case (ii) and regard case (i) as a special case of case (ii). So, we suppose that node x_i receives from node x_{i-1} a QREQ(g_{ini}, y, c, $PATH$) for possibly supporting a query y initiated by node x_0, where g_{ini} is the grid quorum computed by x_0 (by the above step A), c is the cost calculated by x_{i-1}, and $PATH$ is a list of 2-tuples, where each 2-tuple is of the form (*node_id*, *quorum*). Note that $PATH$ contains the nodes that the QREQ has traversed so far and the grid quorums chosen by them. In case that x_i is the query initiator (i.e., $x_0 = x_i$), we will imagine that a virtual QREQ is sent by x_i to itself such that $c = 0$ and $PATH = ()$ is an empty list. On receipt such a QREQ, the following discusses how x_i rebroadcasts this QREQ.

First, x_i will find a quorum to serve query y, which we call $g_{ser}(y)$. If x_i is not currently passed by any query path, it will set $g_{ser}(y) = g_{ini}$. Otherwise, x_i will try to pick an existing quorum in its quorum set $G(x_i)$ or adopt g_{ini} to serve y. It will try to pick an existing one in $G(x_i)$ first. Recall the definition of duty cycle in Eq. 1. Given $G(x_i)$, we can estimate x_i's duty cycle as follows:

$$DTY(G(x_i)) = 1 - \prod_{g \in G(x_i)} (1 - dty(g)). \tag{2}$$

Also, from x_i's QSI, we can measure x_i's current traffic load as follows. For each query z, in x_i's QSI, its load can be calculated by $ld(z) = \frac{len(z)}{r} \cdot \frac{1}{p(z)}$, where $len(z)$ is the length of each sensing report and $p(z)$ is the period per report for query z. So x_i's current traffic load is

$$LD(x_i) = \sum_{\forall z \in QSI\ of\ x_i} ld(z). \tag{3}$$

Then, x_i can measure whether its current quorum set can accommodate y or not by checking $LD(x_i) + ld(y) \leq DTY(G(x_i))$. If so, x_i will try to pick a candidate quorum $g_{can} \in G(x_i)$ with sufficient capacity to serve y. The *capacity of* g_{can} is defined as follows:

$$Cap(g_{can}) = \frac{\sum_{s_j \in QS(g_{can})} \frac{1}{s\text{-}deg(s_j)}}{n_1(g_{can}) \times n_2(g_{can})}, \tag{4}$$

where $n_1(g_{can})$ and $n_2(g_{can})$ are the numbers of rows and columns of g_{can}, respectively, $QS(g_{can})$ means the set of quorum slots of g_{can}, and $s\text{-}deg(s_j)$ is the *share degree* of the quorum slot s_j in g_{can}. Here the share degree of s_j is the estimated average number of quorums which will also regard slot s_j as a quorum slot. This is due to the fact that x_i may be running several quorums simultaneously to support multiple query paths, so quorum g_{can} can only have an equal share of that slot. (For example, in Fig. 2, the share degree of slot 5 of g_2 is two and the share degree of slot 6 of g_2 is one.) If there exists one g_{can} such that

$$Cap(g_{can}) \geq ld(y) + \sum_{z \text{ supported by } g_{can}} ld(z), \tag{5}$$

then g_{can} will be assigned to support y and we will set $g_{ser}(y) = g_{can}$. Otherwise, no existing quorum in $G(x_i)$ can support y and we will check the following two conditions to see if it is possible to include g_{ini} into $G(x_i)$:

- $DTY(G(x_i) \cup \{g_{ini}\}) \geq LD(x_i) + ld(y)$
- $Cap(g_{ini}) \geq ld(y)$

If both conditions are met, we will set $g_{ser}(y) = g_{ini}$; otherwise, this query is beyond the capacity of x_i to be supported and the QREQ will be discarded. Finally, if y can be supported, x_i will append the 2-tuple $(x_i, g_{ser}(y))$ to the list $PATH$ and proceed to the next step.

The above steps have determined the quorum $g_{ser}(y)$ to support y. Next, we will compute the additional energy cost to support y. There are two costs associated with this: (i) the average extra energy cost C_{act} for x_i to remain active per slot and (ii) the average extra energy cost C_{tx} for x_i to transmit data for y per slot. For (i), recall that $g_{ser}(y)$ is the quorum to serve y by x_i. Let $g'_{ser}(y)$ be the quorum selected by x_{i-1} to serve y. We will actually enforce x_i to include $g'_{ser}(y)$ into its quorum set, so that x_i can smoothly transmit data to x_{i-1}. The cost C_{act} is defined as

$$C_{act} = E_{act} \times (DTY(G(x_i) \cup \{g_{ser}(y), g'_{ser}(y)\}) - DTY(G(x_i))),$$

where E_{act} is the energy to remain active for one full slot. This means the extra amount of energy for x_i to remain active per slot in order to support y. For (ii), the cost C_{tx} is defined as

$$C_{tx} = (E_{tx} - E_{act}) \times \frac{len(y)}{r} \times \frac{1}{p(y)},$$

where E_{tx} is the energy to transmit one full slot of data.

The total addition energy cost for x_i to support y is $C_{act} + C_{tx}$. So, we will set $c = c + C_{act} + C_{tx}$.

4.1.3 QREQ Rebroadcasting

The above steps have determined the new c and $PATH$ if x_i decides to support y. Node x_i will also maintain the minimum cost c_{min} for all paths from x_0 to x_i that x_i has learned so far. If $c_{min} \geq c$, then x_i will rebroadcast QREQ($g_{ini}, y, c, PATH$) containing the new c and $PATH$ and set $c_{min} = c$. Note that in cast that x_i is the source s_r, rebroadcasting QREQ is not necessary (this will be discussed in Sect. 4.2).

4.2 Query-Replying Process

When a node x_i receives from x_{i-1} a QREQ($g_{ini}, y, c, PATH$) initiated by a node x_0 and finds that it is the sink node of the query y, it will prepare to periodically report its sensing data to x_0 according to the parameters specified in the query. Node x_i will collect QREQs for a while and choose the QREQ($g_{ini}, y, c, PATH$) with the lowest cost c. Then x_i will unicast QREP($y, PATH$) back to x_0. The QREP will sequentially traverse nodes along the reverse direction of $PATH$.

For each node x_j receiving the QREP, it can identify its serving quorum $g_{ser}(y)$ recorded in the $PATH$. There are two cases:

- If $G(x_j) = \{g_{def}\}$, x_j will directly set $G(x_j) = \{g_{ser}(y)\}$.
- Otherwise, x_j will set $G(x_j) = G(x_j) \cup \{g_{ser}(y)\}$.

Also, x_j can find the serving quorum, say $g'_{ser}(y)$, picked by its previous node in $PATH$. If $g_{ser}(y) \neq g'_{ser}(y)$, x_j will further set $G(x_j) = G(x_j) \cup \{g'_{ser}(y)\}$. This is for x_j to cooperate with its previous node so as to smoothly transmit its data to its previous node. Finally, x_j will adjust its QSI table as follows (refer to Table 1). A new entry will be added such that Query = y, Up_Node = x_j's previous node, Down_Node = x_j's next node, Quorum = $g_{ser}(y)$, and Additional_Quorum = $g'_{ser}(y)$.

After a node adjusts its quorum set, it can wake up and sleep according to the quorums in its set. Quorums do not need to synchronize with each other. Whenever any quorum in its set enters a quorum slot, the node has to be active in that slot. Also note that when a quorum slot belongs to multiple queries, the transmission opportunity should be equally shared by all these queries.

4.3 Query-Removing Process

When a query session y terminates, the sink node can identify this fact by checking its QSI table. Then it can initiate a *QREM(y)* packet along the query path to the sink. Each intermediate node when receiving the *QREM(y)* will remove the corresponding entry from its QSI table. Also, the corresponding quorums to support will be removed from their quorum slots. Again, each node will wake up and sleep according to its new quorum slots.

4.4 Local Slot Synchronization

Although the quorum system can guarantee the communication opportunity of any two asynchronous nodes, in this section we will suggest a lightweight local slot synchronization to improve energy efficiency and reduce transmission delays of sensing reports. Here, we only propose to synchronize local nodes' slots and local nodes' quorums. We summarize our rules as follows:

- At the clock level, two neighboring nodes will try to synchronize their clocks by aligning their slots. That is, they will try to synchronize the beginning of slots at each side.
- At the quorum level, if two neighboring nodes use the same quorum in their quorum sets, they will try to synchronize this quorum by aligning the first slot of this quorum at each side. (Different quorums of these two nodes do not need to be synchronized. Similarly, inside each node, two different quorums do not need to be synchronized).

The above two rules do not address how to break the tie when a node has multiple neighbors and/or when a node shares the same quorum with multiple neighbors. We propose to assign priority by the following rules:

- Along a query path, a node that is closer to the source node has a higher priority.
- Between two query paths, the path which was established earlier (i.e., with an earlier timestamp) has a higher priority.

5 Simulation Results

5.1 Simulation Environments

Since large-scare deployment is difficult to realize, we develop a simulation environment to verify the energy efficiency factor of our cross-layer query-processing protocol. We set up a $400 \times 400\,\mathrm{m}^2$ sensing field, on which hundreds of sensor nodes are randomly deployed. The transmission range and carrier sensing range of each sensor node are set to 50 and 100 m, respectively. In our simulations, we will randomly generate several sink-source continuous query pairs with random report periods and lifetimes. The whole simulation time is 7,200 s. To evaluate the energy consumption, the power consumption rates of a wireless interface are set to 50, 50, 45, and 5 mW under transmit, receive, idle, sleep modes, respectively. The default quorum g_{def} is set to (40, 40, $\{1\}$, $\{1\}$) with each quorum slot fixed to 0.1 s. Hence, each node will initially operate under 5% duty cycle and each quorum group is 160 s.

Figure 3 shows a scenario of our system which runs four continuous queries simultaneously. It shows that there exists path sharing between the sink-source pairs (y_1, y_1') and (y_3, y_3') from node 8 to node 131, and the sink-source pairs (y_2, y_2') and (y_4, y_4') from node 148 to node 24. After the simulation terminates, the percentage of nodes' residual energy is displayed in Fig. 4.

In the following sections, we will discuss the benefit of our cross-layer design and the impact of query loads on our approach.

5.2 Impact of Our Cross-Layer Design

To verify the benefit gained from our cross-layer design, we will compare our approach against two schemes. Both schemes apply shortest path routing. The first one lets each query path adjust its quorum on its own, but there is no coordination between paths' quorums; this scheme is referred to as SP–NC (shortest-path, no-coordination). The second one enforces all quorum paths to share the same quorum; this scheme is referred to SP–GQ (shortest path, global-quorum). We show our results below.

(A) *Comparison with the SP–NC Scheme*: Each query reporting period is set to 60 s. Query requests are randomly injected at a rate of one query per 500 s. Figure 5 shows the minimal residual energy among all nodes. Since our scheme encourages a new path to overlap with existing paths, it shows that the SP–NC scheme is more likely to exhaust some particular nodes' energy.

(B) *Comparison with SP–GQ scheme*: The SP–GQ scheme will pick the quorum with the lowest duty cycle that can meet all nodes' requirement as the global quorum. On the contrary, our scheme can dynamically adjust each query path's quorum. The results are in Fig. 6. We fix the number of nodes to 200 and set the query generation rate to one query per 500 to 1,000 s. It shows that our cross-layer design can result in much higher average residual energy. Even the minimum residual energy of our scheme still significantly outperforms that of SP–GQ. Also, the query generation rate has little impact on the energy consumption of our scheme.

5.3 Impact of Traffic Loads

Recall the query load estimation in Sect. 4.1. It can be influenced by three factors: transmission rate, packet length per report, and reporting period. In the following, we will discuss the impact of traffic loads on energy consumption.

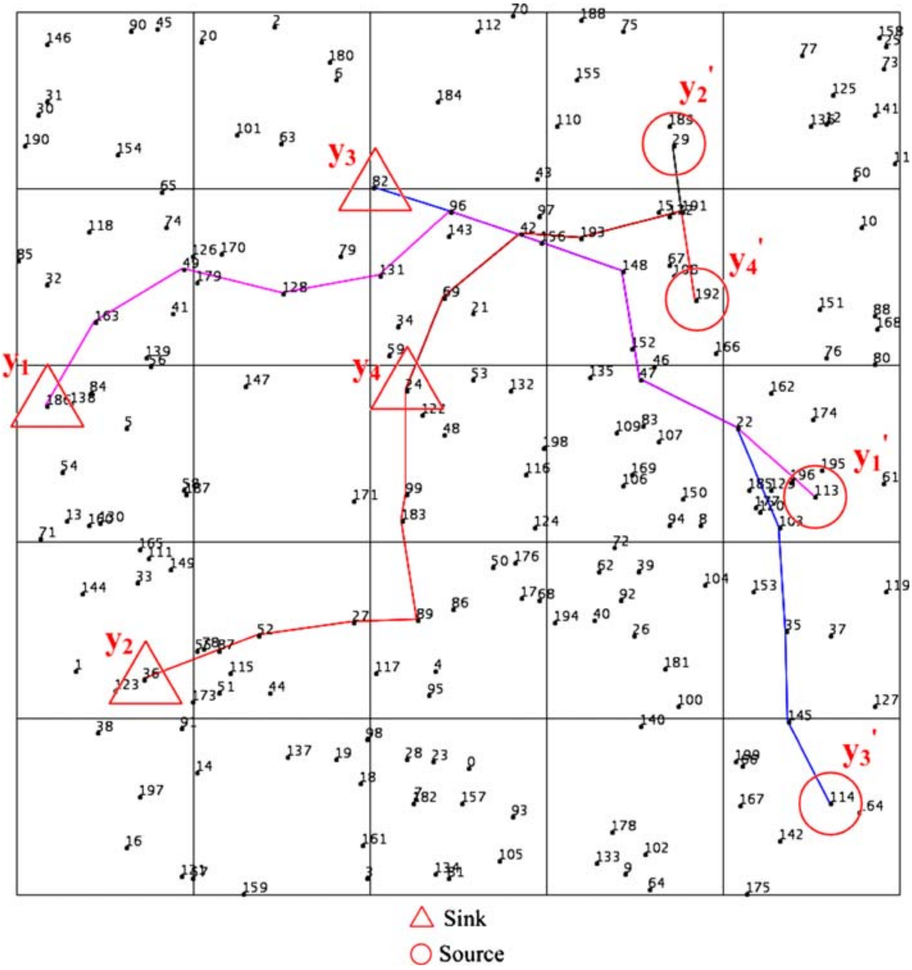

Fig. 3 A path-sharing scenario

(A) *Impact of Transmission Rate*: A smaller transmission rate r will result in slower transmission (and thus a higher traffic load). Hence, we evaluate the energy consumption of our system by varying the transmission rate at 250, 100, 50, and 10 kbps. In Fig. 7a, b, we randomly inject queries at a rate of one query per 1,000 s. In Fig. 7c, d, we randomly inject queries at a rate of three queries per 1,000 s. Each report is 100 bytes. We can see that a lower r might incur higher energy consumption. In Fig. 7a, c, we see that both transmission rate and number of nodes make little impact on the average residual energy because our protocol only causes nodes on query paths to increase their duty cycles. All other nodes still operate with the default quorum. However, if we look at the node with the minimal residual energy, there do exist some differences, as shown in Fig. 7b, d. A lower r will cause some nodes to consume more energy than others but the impact is still quite smaller.

(B) *Impact of Packet Length*: Here, we vary the length len per report to evaluate the energy performance of our scheme. The transmission rate r is fixed to 250 kbps and len varies from 100, 1,000, to 5,000 bytes. Similar with the previous case, the query generation

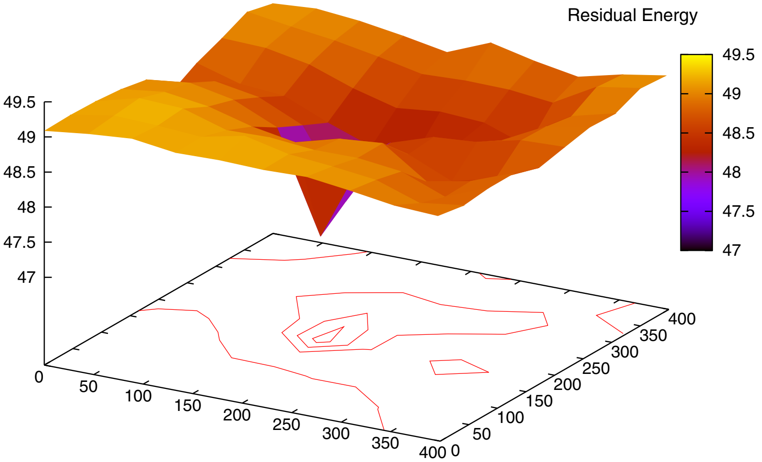

Fig. 4 A scenario of the percentage of nodes' residual energy after executing four continuous queries

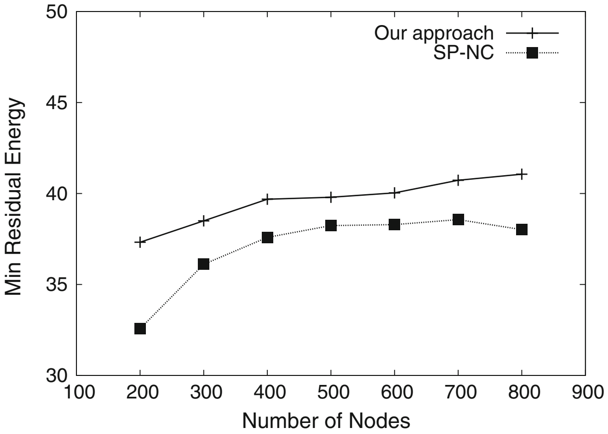

Fig. 5 Comparison to the SP–NC scheme on minimal residual energy

rates are one and three queries per 1,000 s in Fig. 8a–b and Fig. 8c–d, respectively. Figure 8a, c shows that the average residual energy under different $lens$, while Fig. 8b, d shows the minimal residual energy under different $lens$. The tread is generally the same as that in Fig. 7.

(C) *Impact of Query Period*: In this scenario, we set $r = 250$ kbps and $len = 100$ bytes and vary the reporting period p from 30 to 70 s. The query generation rates remain the same with the previous two experiments. The results are similar to the previous cases. As Fig. 9 shows a higher reporting period will incur less energy consumption. From Fig. 9, we see that reporting period (p) has more impact on energy consumption than transmission rate (r) and packet length (len). This is because a lower reporting period will cause nodes to use smaller quorums to serve them. Smaller quorums can easily increase nodes' duty cycles.

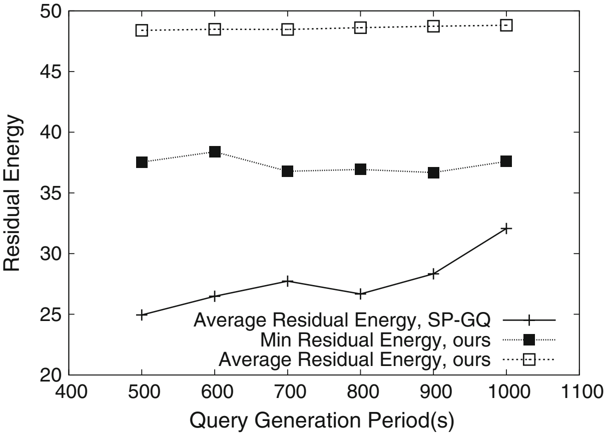

Fig. 6 Comparison to the SP–GQ scheme on nodes' residual energy

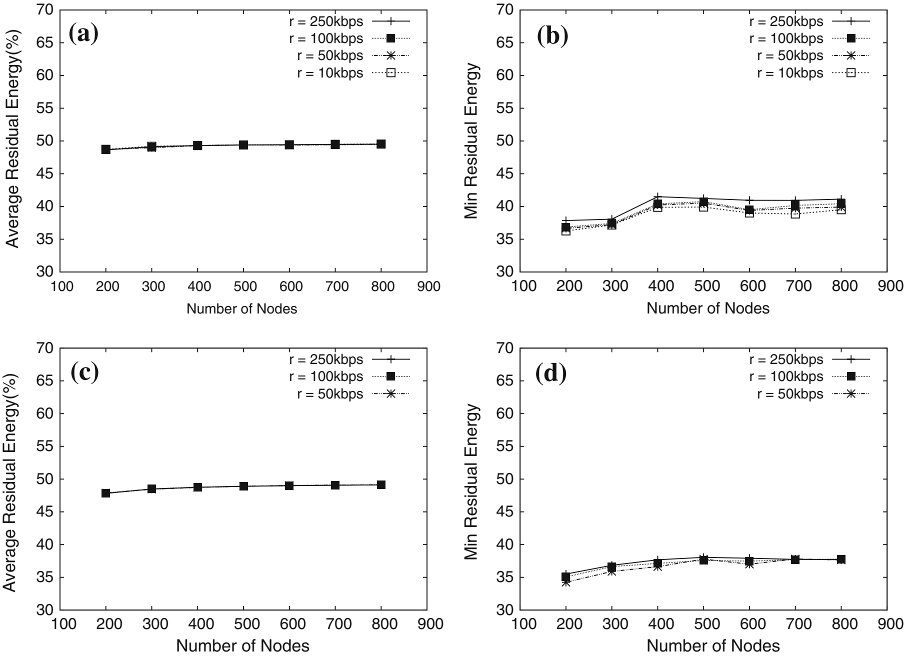

Fig. 7 The energy consumption of our system under different transmission rates (r)

6 Conclusions

We have developed a query-processing protocol to support multiple continuous queries simultaneously in a wireless sensor network. Our design emphasizes on increasing the overlapping of query paths for energy efficiency. It adopts the grid-quorum system and extends it to the

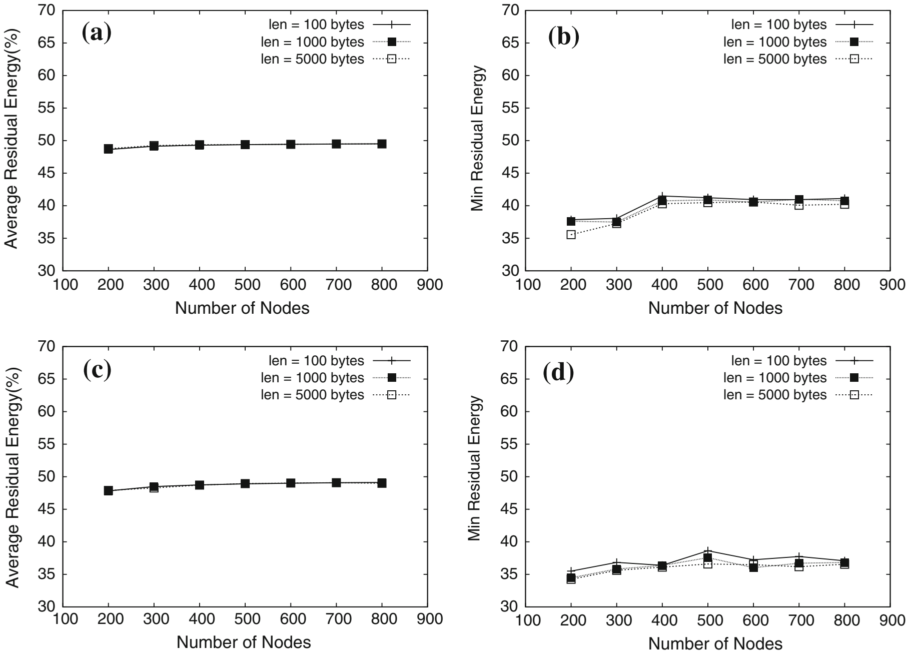

Fig. 8 The energy consumption of our system under different lengths per report (len)

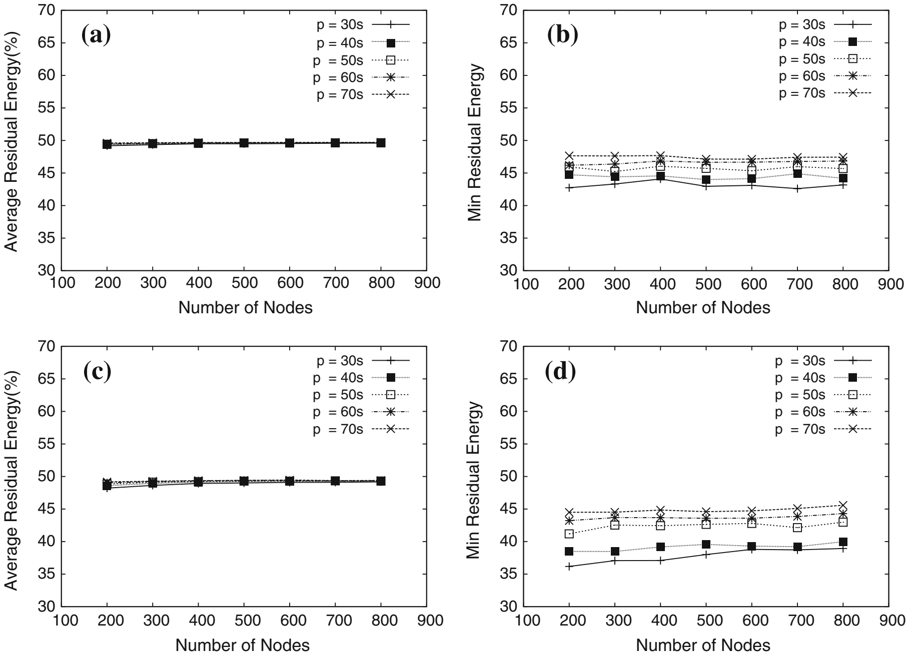

Fig. 9 The energy consumption of our system under different reporting periods (p)

concept of quorum set. We modify the original DSR routing scheme by adding a cost metric to choose quorums along a query path. Simulation results also verify the correctness and performance of the proposed scheme. In the future, we will consider this issue in mobile WSNs.

Acknowledgements Y.-C. Tseng's research is co-sponsored by MoE ATU Plan, by NSC grants 95-2221-E-009-058-MY3, 96-2218-E-009-004, 97-3114-E-009-001, 97-2221-E-009-142-MY3, and 97-2218-E-009-026, by MOEA under grant 94-EC-17-A-04-S1-044, by ITRI, Taiwan, and by III, Taiwan.

References

1. Chao, C.-M., Sheu, J.-P., & Chou, I.-C. (2006). An adaptive quorum-based energy conserving protocol for ieee 802.11 ad hoc networks. *IEEE Transactions on Mobile Computing, 5*(5), 560–570.
2. Chen, B., Jamieson, K., Balakrishnan, H., & Morris, R. (2001). SPAN: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks. In *Proceedings of ACM international conference on mobile computing and networking (MobiCom).*
3. Du, S., Saha, A. K., & Johnson, D. B. (2007). RMAC: A routing-enhanced duty-cycle MAC protocol for wireless sensor networks. In *Proceedings of IEEE INFOCOM.*
4. Fan, K.-W., Liu, S., & Sinha, P. (2008). Dynamic forwarding over tree-on-dag for scalable data aggregation in sensor networks. *IEEE Transactions on Mobile Computing, 7*(10), 1271–1284.
5. He, T., Huang, C., Blum, B. M., Stankovic, J. A., & Abdelzaher, T. (2003). Range-free localization schemes for large scale sensor networks. In *Proceedings of ACM international conference on mobile computing and networking (MobiCom)*, pp. 81–95.
6. Heinzelman, W. R., Chandrakasan, A., & Balakrishnan, H. (2000). Energy-efficient communication protocols for wireless microsensor networks. In *Proceedings of Hawaii international conference on systems science (HICSS).*
7. Hu, P., Hong, P.-L., Li, J.-S., & Qin, Z.-Q. (2006). TAP: Traffic-aware topology control in on-demand ad hoc networks. *Computer Networks, 29*(18), 3877–3885.
8. Huang, C.-F., Tseng, Y.-C., & Lo, L.-C. (2007). The coverage problem in three-dimensional wireless sensor networks. *Journal of Interconnection Networks, 8*(3), 209–227.
9. Huang, X.-M., & Ma, J. (2006). Optimal distance geographic routing for energy efficient wireless sensor networks. *International Journal of Ad Hoc and Ubiquitous Computing, 1*(4), 203–209.
10. Intanagonwiwat, C., Govindan, R., Estrin, D., Heidemann, J., & Silva, F. (2003). Directed diffusion for wireless sensor networking. *IEEE/ACM Transactions on Networking, 11*(1), 2–16.
11. Jiang, J.-R., Tseng, Y.-C., Hsu, C. S., & Lai, T.-H. (2005). Quorum-based asynchronous power-saving protocols for IEEE 802.11 ad hoc networks. *ACM/Kluwer Mobile Networks and Applications, 10*(1/2), 169–181.
12. Kochhal, M., Schwiebert, L., & Gupta, S. (2003). Role-based hierarchical self organization for wireless ad hoc sensor networks. In *Proceedings of ACM international workshop on wireless sensor networks and applications (WSNA).*
13. Li, Q., DeRosa, M., & Rus, D. (2003). Distributed algorithm for guiding navigation across a sensor network. In *Proceedings of ACM international symposium on mobile ad hoc networking and computing (MobiHoc)*, Maryland, USA.
14. Liao, W.-H., Wang, H.-H., & Wu, W.-C. (2007). An adaptive MAC protocol for wireless sensor networks. In *Proceedings of IEEE international symposium on personal, indoor and mobile radio communications (PIMRC).*
15. Madden, S., Franklin, M. J., Hellerstein, J., & Hong, W. (2002) TAG: A tiny aggregation service for ad-hoc sensor networks. In *Proceedings of ACM international symposium on operating systems design and implementation.*
16. Meguerdichian, S., Koushanfar, F., Potkonjak, M., & Srivastava, M. B. (2001). Coverage problems in wireless ad-hoc sensor networks. In *Proceedings of IEEE INFOCOM.*
17. Pan, M.-S., Yeh, L.-W., Chen, Y.-A., Lin, Y.-H., & Tseng, Y.-C. (2008). A WSN-based intelligent light control system considering user activities and profiles. *IEEE Sensors Journal, 8*(10), 1710–1721.
18. Park, H., Srivastava, M. B., & Burke, J. (2007). Design and implementation of a wireless sensor network for intelligent light control. In *Proceedings of ACM/IEEE international conference on information processing in sensor networks (IPSN).*

19. Ratnasamy, S., Karp, B., Shenker, S., Estrin, D., Govindan, R., Yin, L., & Yu, F. (2003). Data-centric storage in sensornets with GHT, a geographic hash table. *Mobile Networks and Applications, 8*(4), 427–442.
20. Savvides, A., Han, C.-C., & Strivastava, M. B. (2001). Dynamic fine-grained localization in ad-hoc networks of sensors. In *Proceedings of ACM international conference on mobile computing and networking (MobiCom)*, pp. 166–179.
21. Schaffer, P., & Vajda, I. (2007). CORA: correlation-based resilient aggregation in sensor networks. In *Proceedings of ACM/IEEE international symposium on modeling, analysis and simulation of wireless and mobile systems (MSWiM).*
22. Skordylis, A., Guitton, A., & Trigoni, N. (2006). Correlation-based data dissemination in traffic monitoring sensor networks. In *Proceedings of IEEE wireless communications and networking conference (WCNC).*
23. Sohrabi, K., Gao, J., Ailawadhi, V., & Pottie, G. J. (2000). Protocols for self-organization of a wireless sensor network. *IEEE Personal Communications, 7*(5), 16–27.
24. Szewczyk, R., Mainwaring, A., Polastre, J., Anderson, J., & Culler, D. (2004). An analysis of a large scale habitat monitoring application. In *Proceedings of ACM international conference on embedded networked sensor systems (SenSys).*
25. Tseng, Y.-C., Hsu, C.-S., & Hsieh, T. Y. (2003). Power-saving protocols for IEEE 802.11-based multi-hop ad hoc networks. *Computer Networks, 43*(3), 317–337.
26. Tseng, Y.-C., Pan, M.-S., & Pan, M.-S. (2006). A distributed emergency navigation algorithm for wireless sensor networks. *IEEE Computer, 39*(7), 55–62.
27. Wu, M., Xu, J., & Tang, X. (2006). Processing precision-constrained approximate queries in wireless sensor networks. In *Proceedings of ACM/IEEE international conference on mobile data management*.
28. Wu, T.-T., & Ssu, K.-F. (2005). Determining active sensor nodes for complete coverage without location information. *International Journal of Ad Hoc and Ubiquitous Computing, 1*(1/2), 38–46.
29. Xu, Y., Heidemann, J., & Estrin, D. (2001). Geography-informed energy conservation for ad hoc routing. In *Proceedings of ACM international conference on mobile computing and networking (MobiCom).*
30. Ye, W., Heidemann, J., & Estrin, D. (2002). An energy-efficient MAC protocol for wireless sensor networks. In *Proceedings of IEEE INFOCOM*.
31. Zheng, T., Radhakrishnan, S., & Sarangan, V. (2005) PMAC: An adaptive energy-efficient MAC protocol for wireless sensor networks. In *Proceedings of IEEE international parallel and distributed processing symposium (IPDPS).*

Author Biographies

Chia-Hung Tsai received his B.S. and M.S. degrees in Computer Science and Information Engineering from the National Chiao-Tung University, Taiwan, in 2004 and 2006, respectively. He is currently pursuing Ph.D. in the Department of Computer Science, National Chiao-Tung University, Taiwan. His research interests include wireless communication and sensor networks.

Tsu-Wen Hsu received his B.S. and M.S. degrees in Computer Science from National Chiao-Tung University, Taiwan, in 2006 and 2008, respectively. His research interests include wireless networks, sensor networks, and resource management.

Meng-Shiuan Pan received the B.S. and M.S. degrees from the National Chung Cheng University and National Tsing Hua University, Taiwan, in 2001 and 2003, respectively. He obtained his Ph.D. in the Department of Computer Science, National Chiao Tung University, Taiwan, in 2008. His research interests include mobile computing and wireless communication.

Yu-Chee Tseng obtained his Ph.D. in Computer and Information Science from the Ohio State University in January of 1994. He is Professor (2000–present), Chairman (2005–present), and Associate Dean (2007–present) at the Department of Computer Science, National Chiao-Tung University, Taiwan. He is also Adjunct Chair Professor at the Chung Yuan Christian University (2006–present). Dr. Tseng received the Outstanding Research Award, by National Science Council, ROC, in both 2001–2002 and 2003–2005, the Best Paper Award, by International Conference on Parallel Processing, in 2003, the Elite I. T. Award in 2004, and the Distinguished Alumnus Award, by the Ohio State University, in 2005. His research interests include mobile computing, wireless communication, and parallel and distributed computing. Dr. Tseng serves on the editorial boards for Telecommunication Systems (2005–present), IEEE Trans. on Vehicular Technology (2005–2009), IEEE Transactions on Mobile Computing (2006–present), and IEEE Transactions on Parallel and Distributed Systems (2008–present).