Telecommun Syst (2013) 54:405–415
DOI 10.1007/s11235-013-9742-x

An energy-aware grid-based routing scheme for wireless sensor networks

Yuan-Po Chi · Hsung-Pin Chang

Published online: 19 July 2013

Abstract As an important field of emerging technology, wireless sensor networks (WSN) offer many new possibilities for applications such as target tracking and environmental surveillance by allowing the observer to move around freely. However, disseminating sensing data to the mobile observer raises significant design challenges for the routing scheme. In addition, WSN often operate under certain energy constraints, and therefore reducing energy dissipation in order to prolong the lifetime of the WSN is another challenge that must be faced. Most proposed routing protocols focus on achieving effective data dissemination and energy efficiency at the same time as working to satisfy the requirements of the mobile observer. However, almost all of these methods use frequent rerouting as a way of handling the mobility issue. Such rerouting increases both overheads and energy consumption, resulting in a trade-off between the need for rerouting to optimize network operations and that of maximizing network lifetime. This paper presents the Energy-aware Grid-based Routing Scheme (EAGER) for WSN with mobile observers, which is an approach that seeks to save more energy in the context of dynamic topology. In this paper, EAGER is compared to other proposed grid-based schemes by using extensive simulations. These simulations clearly show that EAGER outperforms other grid-based schemes in terms of both energy efficiency and routing performance.

Keywords Wireless sensor networks · Routing · Energy-aware · Virtual grid-based · Mobile sink

Y.-P. Chi (✉)
Department of Computer Science and Engineering, National Chung-Hsing University, Taichung, Taiwan
e-mail: phd9404@cs.nchu.edu.tw

H.-P. Chang
Department of Computer Science and Engineering with Institute of Networking and Multimedia, National Chung-Hsing University, Taichung, Taiwan
e-mail: hpchang@cs.nchu.edu.tw

1 Introduction

Recent technological advances now make it possible to integrate micro-electromechanical systems, micro-sensors, and wireless communication devices into miniature, low-cost, low-powered sensor nodes. Comprising large numbers of sensor nodes that allow the observer to move around freely, wireless sensor networks (WSN) offer many new possibilities for application in areas such as target tracking [1, 20, 21, 28, 32] and environmental observation [17]. For example, in military target tracking and surveillance, soldiers need to be able to move in various directions at any given time in order to monitor the movements of enemy tanks [23]. Likewise, in wild animal enclosures, park administrators frequently need to move around in order to monitor animal behavior.

Figure 1 illustrates the architecture of a WSN. First, a number of sensor nodes are deployed, either by pre-planning or by dropping them from a vehicle, within the monitored area. These allow the observer to send relevant monitoring commands in order to query any of the specified targets. As soon as one or more of the sensor nodes senses stimuli from the target within the monitored area, one of these nodes (known as the "source") will immediately report the relevant data back to the observer (known as the "sink") via a wireless channel. Using a wireless mobile device or laptop, the observer can receive data disseminated from the source (such as location and temperature) and the data can also be

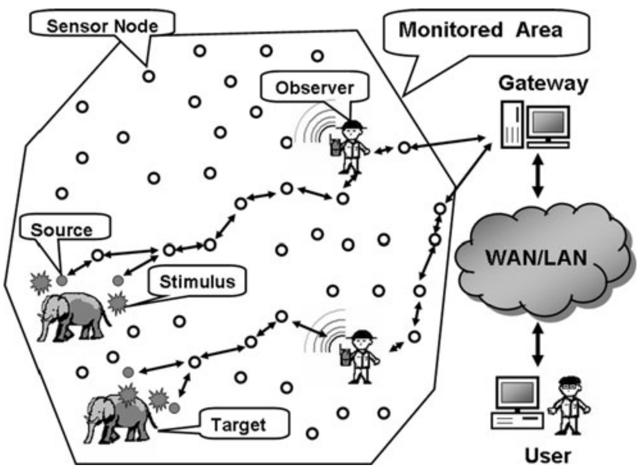

Fig. 1 Architecture of a WSN

sent to other users for further analysis and data mining. Consequently, the disseminating of sensing data from the source to the sink is a fundamental function of WSN.

Nonetheless, WSN operate independently of any existing infrastructure [5]. They inherit without the addressing feature normally used in routing schemes for traditional wired networks. Moreover, the routing protocol for WSN needs to be capable of data aggregation, data dissemination, and special application—and therefore using the proposed routing protocols for wireless ad hoc networks is also not suitable for WSN [24]. Therefore, finding an efficient method of disseminating the sensing data and query commands between the source and the mobile sink presents a significant challenge for the design of WSN routing schemes [2, 7, 30].

In addition to this challenge, WSN are often operated with strict energy constraints since the sensor nodes are battery-operated and therefore resource-limited. Reducing energy dissipation to prolong the life of WSN is another important design issue for the routing scheme. The majority of proposed WSN routing schemes tend to focus on realizing both efficient data dissemination and energy consumption [10, 11, 25]. However, in some WSN applications, the observer needs to be able to move in any number of directions in order to be able to track multiple targets (such as wild animals or enemy vehicles). Since the observer could potentially move to any given location within the WSN at any given time, propagating the sensing data from tracked targets from the source to the mobile sink poses yet another challenge to the design of routing protocol [2]. In terms of handling mobile sinks, some proposed schemes seek to resolve this issue by using rerouting [8, 13, 24, 27] or relaying agent approaches [5]. However, in such a scenario, frequent movement of the sink would lead to either frequent rerouting or long relaying chains, which would necessarily increase both operating overheads and energy consumption. Therefore, it is clear that there is a trade-off between reducing energy consumption to prolong network lifetime and rerouting to maintain network topology.

In this paper, we propose a grid-based routing scheme, called Energy-aware Grid-based Routing (EAGER), to disseminate data between the target and multiple mobile sinks in order to prolong the lifetime of the network. In order to achieve energy efficiency in the context of dynamic topology, EAGER uses a rerouting method to reduce rerouting frequency and also a time-scheduling method to manage the energy consumption of the grid. The remainder of this paper is organized into four sections. Section 2 reviews related research and assumptions; Sect. 3 details the architecture of EAGER; Sect. 4 presents the simulation results; and, finally, Sect. 5 comprises the conclusions and future work of this paper.

2 Related work

Many WSN routing schemes have been proposed to date. In general, routing schemes are classified into three categories based on network structure, namely: flat, hierarchical, and location-based routing [2, 31]. The hierarchical-based schemes aim to cluster the nodes so that cluster heads can be responsible for aggregating and disseminating data. The grid-based protocols inherit the character of hierarchical-based schemes and use a virtual grid structure to cluster the nodes with the aim of saving energy [3, 8, 13, 14, 24, 27, 33]. TTDD [8] provides two disseminating tiers (i.e., high and low) for large-scale sensor networks with multiple mobile sinks. Once a sensor node becomes a source, it broadcasts an event-announcement message for constructing a virtual grid structure to cover the entire network. The nodes close to the cross points of the virtual grid structure form the high tier and act as data dissemination points. In contrast, the low tier is comprised of the paths from each sink to the closest local dissemination point. Using both the high and low tiers, TTDD is able to propagate data from the source to the sink. However, the dynamic grid construction of TTDD is energy intensive, especially as the number of sources increases [5].

GMDQP [5] improves on TTDD by eliminating the overhead costs of grid construction. Instead of building grid structures from multiple sources, GMDQP builds the grid structure from the sink side. When a sink first queries a target, it chooses a close sensor node as the primary data examination node (PDEN) in order to initiate the grid construction. PDEN first calculates the position of four adjacent cross points (DEP) on the grid. Then, PDEN floods the sink's query message using the greedy geographical forwarding method [9] to elect four new data examination nodes (DEN) near the DEPs. Meanwhile, each DEN repeats this action to elect adjacent new DENs. Finally, a grid structure rooted at the PDEN is built, with each DEN caching the position of the upstream DEN for routing. As the source reports the sensing data to the sink, it reaches out to a close DEN in order to deliver the data. Therefore, based on the routing information of each DEN, the data can be delivered to the sink. Also, GMDQP applies a mobile agent scheme to resolve the dissemination issue of mobile sinks. When the sink moves, the sink selects a neighbor as the new mobile agent. Both the PDEN and existing mobile agents keep the position of the new mobile agent for forwarding data. Thus, maintenance of the position of the mobile agents is required, which means that maintenance costs will be high for sinks with high mobility.

Similarly, CODE [13] also relies on the grid structure and revises the GAF protocol [26] to establish data dissemination paths between the source and mobile sink. CODE selects a coordinator to disseminate data within each grid cell. As the source detects the target's stimulus, it floods a message containing its location to inform all coordinators before reporting the data. Using this informed location, the sink then builds a routing path toward the source by forwarding a data query message. In addition, each coordinator applies the informed location with its own location to obtain an upstream coordinator for routing. Furthermore (and as is also the case for GMDQP), CODE adopts the agent approach to solve the problem of sink mobility. First, the sink chooses the closest coordinator as its agent. When the sink moves, it polls nearby coordinators to choose a new agent. When the sink moves from the grid cell where it has stayed previously, again it elects a new agent instead of using the old agent. The new agent is responsible for rebuilding the new routing path and notifying the old agent to remove the obsolete routing path. Consequently, these rebuilding operations create additional overheads and have a number of other shortcomings. If the sink moves quickly, the new agent must perform the rebuilding process frequently, which necessarily incurs significant overheads. Also, while the source reports data during the rebuilding period, there is the possibility that data delivered along the obsolete routing path will be lost.

As with CODE, EADA is based on the GAF protocol in the sense that it retains some sensor nodes to participate in network processing to prolong the network lifetime [24]. In addition to this, EADA also confines the forwarding area of query messages sent by the sink with a fan-as zone (as is shown in Fig. 2) to avoid broadcast-storm issues. With regards to the handling of sink mobility, EADA exploits the CODE protocol by using the fan-as zone. EADA applies a fan-as zone between the sink and the source to confine the forwarding area of query messages sent by the sink. Using such a confined zone, EADA limits the forwarding number of query messages in order to eliminate rerouting overhead costs while handling sink mobility. However, if multiple mobile sinks move within the monitored area, this rerouting approach may lead to additional communication overheads.

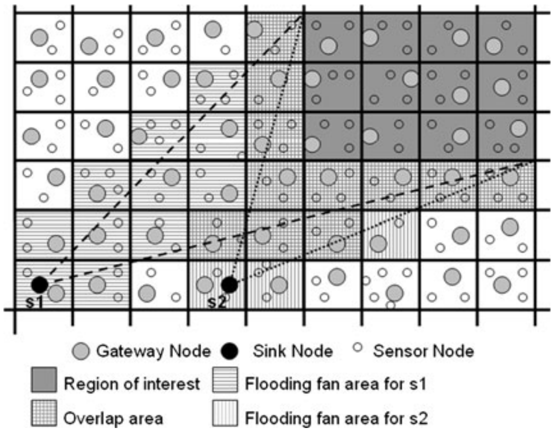

Fig. 2 Overlap area for multi-sinks in EADA

While all sinks send their own query messages to the same source, all fan-as zones for forwarding messages will overlap (as is shown in Fig. 2). All coordinators within the overlap area will need to relay many messages, meaning that they will consume energy quickly. If some mobile sinks move quickly, the energy consumption of the affected coordinators will greatly affect the overall network lifetime.

As with EADA, the proposed EAGER is a grid-based routing protocol that uses only a small number of sensor nodes to participate in the network processing. EAGER also considers a new rerouting approach to resolve the mobility issue of multiple sinks. Furthermore, EAGER utilizes a time-scheduling method to keep idle nodes dormant in order to reduce unnecessary energy consumption.

3 The energy-aware grid-based routing scheme

Before presenting the EAGER scheme, we must make a number of assumptions. First, it is assumed that the echo sensor node is aware of its own location via a GSP receiver or other location estimation technique [3, 14, 29]. Alternatively, all sensor nodes are homogeneous and the built-in time clocks of all nodes are synchronized before deployment. After being deployed, all nodes are stationary and capable of sensing stimuli generated from the targets via the sensor channel. When multiple sensor nodes detect target stimuli simultaneously, the node that detects the greatest signal strength becomes the source and generates a data report to all sinks via the wireless channel. Each sink is capable of collecting the target data from the monitored area at any time.

3.1 Construction of the virtual-grid structure

At startup, EAGER divides the monitored area into a number of virtual grid cells, such as is shown in the example in Fig. 3.

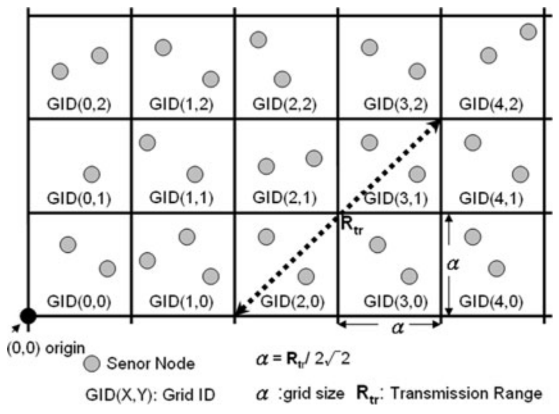

Fig. 3 Construction of the virtual-grid structure

A unique pair of numbers, known as the Grid Identification (GID), is used to identify each grid cell. All sensor nodes located in the same grid cell share the same GID. The location (x_0, y_0) indicates the geographic position of the origin of the monitored area. Before the sensor nodes are deployed, both the origin and grid size, α, are set as built-in system parameters for the sensor nodes. Furthermore, the grid size α, which is determined by the transmission range R_{tr}, is defined as $\alpha = R_{tr}/2\sqrt{2}$, allowing it to communicate directly with its eight adjacent grid cells via radio channels. After deployment, each node calculates the GID of the grid cell to which it belongs with its geographic coordinates (X, Y) using (1), where $\lfloor k \rfloor$ is the largest integer less than k.

$$GID(X,Y) = \left\{ (X,Y) | X = \left\lfloor \frac{x - x_0}{\alpha} \right\rfloor, Y = \left\lfloor \frac{y - y_0}{\alpha} \right\rfloor, \right. \\ (x_0, y_0) \in origin; \\ \left. (x_0 \le x) \wedge (y_0 \le y); \alpha : grid\ size \right\} \quad (1)$$

3.2 Election of the grid head

In each grid cell, all members elect a coordinator—called the Grid Head (GH)—to be responsible for disseminating data and managing all members. For the purpose of energy efficiency, all other nodes turn their radios off until they detect the targets' stimulus via their own opening sensor channels. The election of the GH follows the first-mover rule, which is described as follows: First, each node invokes a timer with random intervals and then broadcasts an election packet with its GID. If the node makes an election attempt before it receives an election packet from any other member, this node becomes the GH. Alternatively, if the node receives an election packet before the timer fires, the attempt will be canceled immediately. Once the GH has been elected, it will broadcast a hello packet with its GID to all members and all GHs in adjacent cells. On receiving this hello packet, the members will turn off their radios periodically and keep only their sensing channels active until they sense a stimulus from a target. Meanwhile, if the adjacent GHs receive the hello packet, they will record the GID of the sender of this packet in their neighboring list.

However, for query management, all grid members must turn on their radio channels periodically and the period is set by the GH in the election-completed packet. For instance, if the sink wants to collect a new type of sensing data different from the previous one, it soon delegates its GH to flood a query management packet. Especially, for reducing energy consumption, only the GH is responsible for receiving and forwarding the management packet. As receiving such packet, each GH caches this query until its entire grid members turn on their radio channel. Each GH then sends the query management packet to all of its members, and eventually, all sensor nodes change to monitors the new type of sensing data according to the received new query.

3.3 Time-scheduling method for the grid head

As mentioned in Sect. 3.2, each grid cell elects one node as the GH responsible for disseminating data. As not all GHs will be participating in data dissemination at all times, idle GHs will still be consuming energy and will therefore cause a reduction in the lifetime of the network. To tackle this problem, EAGER unveils a time-scheduling method that allows a number of the idle GHs to be asleep at any given time—with all idle GHs turning off at some point as their radios go to sleep periodically according to assigned sleeping periods.

The detailed mechanics of this method are as follows: First, once a node has been elected as the GH, this node will determine whether to keep its radio active or inactive, based on the sum of the x- and y-coordinates of its GID. If the sum is even, the GH will keep its radio active. Otherwise, the radio will be turned off to allow the GH to sleep for a specified time interval decided by a built-in scheduling method that assigns the sleeping period. This method then divides a time unit into 2^n timeslots and assigns a fixed timeslot number calculated by (2) for sleeping, where GID.X and GID.Y indicate the x- and y-coordinates of the GID, the exponent n is greater than zero.

$$\text{Time slot number} = \left[(\text{GID}.X + \text{GID}.Y) \mod 2^n \right. \\ \left. + \text{GID}.X \mod 2^{(n-1)} \right] \quad (n > 0) \quad (2)$$

For instance, if time unit is divided into four (2^n, n equals to 2) timeslots, where the GID of a GH is $(1, 2)$, its scheduling method will assign it the fourth timeslot for sleeping. The GH will therefore turn the radio on for the first three timeslots, and then turn it off for the fourth timeslot. However, the turning-off action will be suspended in the case

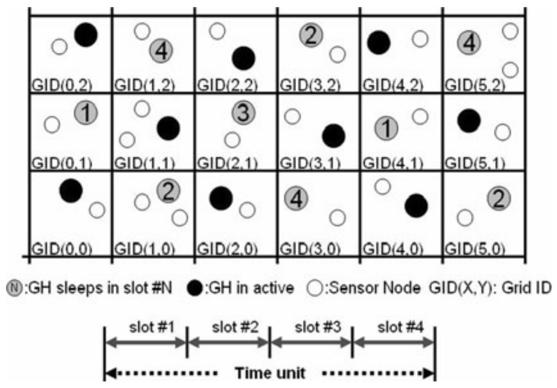

Fig. 4 An example of time scheduling for GHs using four timeslots

that the GH receives data packets and/or detects radio signals sent from adjacent GHs or members of the local grid cell prior to entering sleeping mode. Otherwise, the GH will periodically turn off the radios at the assigned timeslot.

Since each sensor node is homogeneous and its clock synchronized (as mentioned in the assumptions made at the beginning of Sect. 3), the timing schedules for all of the nodes will always be identical prior to deployment. Using such time-scheduling method ensures that the radio channels of any four adjacent GHs are available at any time, though without creating any void communication channels.

Even if the data arrives while the destination GH is in the sleeping period, one of those adjacent GHs whose radios are active will be delegated by the upstream GH for delivering this data instead. Before delivering the data, the upstream GH first checks the scheduling of the destination GH by using (2) with that's GID. Finding the destination GH is in the sleeping period, the upstream GH subsequently sends the data to one of the adjacent GHs whose radio is always available instead.

In Fig. 4, using four timeslots, the GHs colored in gray are labeled by a timeslot number to indicate the timeslot in which they are assigned to sleep. For example, using (2), the assigned sleeping timeslot for the GH located in GID(0, 1) will be the first timeslot. This timeslot is then excluded from those available to any of the adjacent GHs, such as that for the GHs located in GID(0, 1) and GID(2, 1). Using the time-scheduling method not only serves to save energy for the GHs, but also helps to ensure that the radio channels of all GHs are always available in any set of four adjacent cells.

3.3.1 *Determining the exponent n of the time-scheduling equation*

As mentioned above, the time scheduling method utilities (2) to calculate the number of timeslots with the x- and y-coordinates of all GH nodes assigned with the sleeping period. In this equation, the exponent n determines how many time slots can be divided and how many GHs can be scheduled with the same timeslot number. As the exponent n is increased, the interval of the sleeping period is decreased to close to that of without any sleeping scheduling. In this situation, thus, applying the time-scheduling gains less energy saving. In contrast, as the n is decreased, the interval of sleeping period is increased. While n, which equals to one, is minimum, two diagonal GHs of each grid unit (2×2 grid cells) will be assigned with the same sleeping period. Thus, another two adjacent GHs having active radios will be always delegated for delivering to increase the delegation cost.

3.4 Establishing an initial routing path

This section describes a key feature of the EAGER scheme. Before disseminating data, EAGER builds the initial routing paths from source to sink upon the receipt of the data announcing message by means of a simple request-reply operation. Once a node becomes the source, it will report the sensing data to its local Grid Head (LGH), which shares an identical GID with this node. The LGH will first check whether it has a routing path to the sink and, if such a path exists, the LGH will disseminate the data directly. If not, it will flood a route request packet (REQ) to inform the source location and to find the accurate path to the sink. The GH is responsible for relaying the REQ packet, and the REQ packet contains three added fields, namely: the identification number, hop count and visited list. Initially, the value of the hop count is zero and the list is empty. When receiving the REQ, the first step of the GH is to examine the identification number. Having received a REQ, the GH will discard this packet to avoid the delivery loop. Otherwise, the GH will increase the hop count of this packet by one and append the visited list of that with its GID before flooding this packet. On receiving the REQ, the GH caches one REQ packet with the smallest hop number value. If the sink wishes to inquire about the sensing data, it sends a query packet to its LGH.

Subsequently, the sink's LGH will then check whether the route path exists. In the case that the path does not exist, this LGH will send back a route reply packet (REP) with the reverse visited path of its cached REQ to establish the initial route path. On receiving the REP packet, each intermediate GH along the visited path builds the entry of its routing information table (RIT) by the contents of the visited list. Each RIT entry is organized by a tuple of (*destination*, *next*, *previous*, *start*), where *destination* is the GID of the sink's current LGH; *next* is the downstream hop for routing the sensing data to sink node; *previous* is the downstream hop for delivering subsequent monitoring commands sent by sinks towards *start*; and *start* is the GID of the source's LGH. Once the source's LGH has received the REP packet,

Table 1 Pseudo code for determining the next hop

```
Relaying_NextHop ( Starting_Node sn, Ending_Node en )
begin
  // Nexthop.GID(x,y): the GID of the Next hop
  X.offset = en.GID(x) – sn.GID(x)
  Y.offset = en.GID(y) – sn.GID(y)
  Direction(x) = ( X.offset != 0) ? X.offset/| X.offset |: 0
  Direction(y) = ( Y.offset != 0) ? Y.offset/| Y.offset |: 0
  Nexthop.GID(x, y) = GH.GID(x, y) + Direction(x, y)
  if ( lookup_neighbor ( Nexthop ) != null )
    return Nexthop.GID(x,y)
  else
    return NULL
end
```

the establishment of the initial route path is complete. Consequently, using this initial path, the source's LGH is able to disseminate the sensing data straight away.

3.5 Handling sink mobility

This subsection presents the method used by EAGER with regards to handling the mobile sink. As shown in Fig. 5a, the mobile sink is location-aware and periodically checks its current location. If the sink finds that it is in the same grid cell as during the last check, it does nothing. Otherwise, it will broadcast an INFORM-LOC packet containing its current and previous GIDs. On receiving the INFORM-LOC packet, the current sink's LGH is then responsible for sending a BUILD-ROUTE packet to attach the old routing path. The BUILD-ROUTE packet contains the GIDs of two endpoints that include the sink's previous LGH as the ending point and the LGH as the starting point. The next hop for relaying is determined by the Relaying_Nexthop algorithm, which is used in CODE [13], as shown in Table 1. Using this algorithm, the LGH can first obtain a direction close to the ending point. When the next hop receives this packet, it uses the relevant GID as the new starting point to compute the next hop for relaying. Once the sink's previous LGH receives this packet, it will reply a BUILD-REPLY packet with the reverse relaying path. Receiving this BUILD-REPLY packet, the intermediate GHs will insert a new routing entry into their built-in RIT table. Once the LGH has received the BUILD-REPLAY packet and updated its RIT, the establishment of the new routing path for attachment to the original one is complete.

Using the example in Fig. 5a, when the sink moves from Cell-1 to Cell-2, it will broadcast the INFORM-LOC packet in order to build the routing path. Once the INFORM-LOC packet has been received, the LGH of Cell-2 (node F), then sends the BUILD-ROUTE packet to the previous GH (node E). Having received the BUILD-ROUTE packet, node

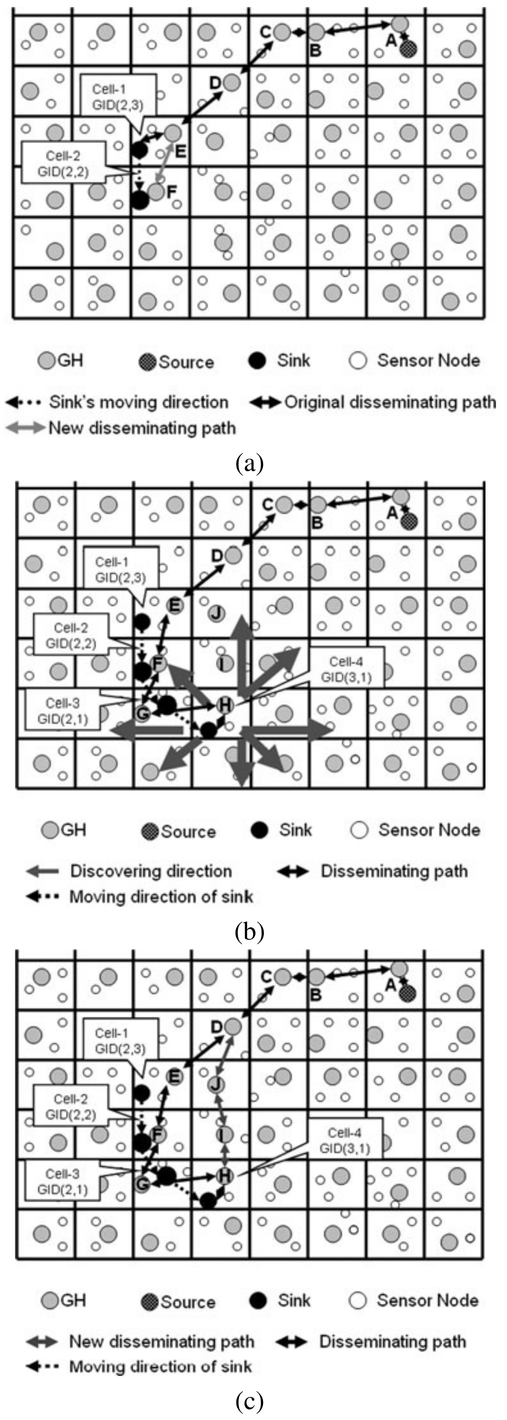

Fig. 5 (**a**) Attaching the disseminating path. (**b**) Discovering a possible rerouting direction. (**c**) Rerouting

E will then update the destination and the subsequent fields of its RIT with the GID of this packet's sender, and will then reply with a BUILD-REPLAY packet. Eventually, on receiving the BUILD-REPLY packet, node F will insert a new entry as (F,null,E,A) into its RIT to establish a new dissemination path (E,F) as is marked by the gray arrow.

Table 2 Pseudo code for discovering

```
Discovering_new_path ( GH, packet )
begin
  // packet: the data packet that has received by GH node
  // candidate: a node that is the destination of rerouting
  max = 0
  candidate = null
  for node in packet.visited_list
  do
    for direction in ( North, Northeast, East, Southeast, South,
                       Southwest, West, Northwest )
    do
      if ( node on the direction )
      then
        if ( distance ( node, GH ) > max )
        then
          max = distance ( node, GH )
          candidate = node
        endif
      endif
    done
  done
  if ( find candidate in the neighbor list )
    return null
  else
    return candidate
  endif
end
```

However, random movements by the sink may cause a curved and lengthy routing path between the source and the sink, leading to higher energy costs for delivering data. Therefore, it is necessary to reroute the path to reduce the cost of data delivery. For the purposes of rerouting, the data packet has one added list—called the visited list—which records the GIDs of all the GHs that deliver data packets. When receiving the data packet, the sink's LGH applies the algorithm shown in Table 2 to examine the visited list of this packet and to see whether a possible new routing path exists among the eight directions discovered. Using the algorithm, the sink's LGH checks whether any of the visited GH nodes are located on any of the eight directions of its position. If any such nodes are discovered, the LGH will choose the furthest GH nodes and send a simple REROUTE packet (containing the rerouting direction) towards those nodes in order to build a new path. Before sending the REROUTE packet, the GH first determines the GID of the next hop by calculating its GID with the rerouting direction. Then, the GH searches its neighboring GH list for this next hop. The absence of the next hop indicates a void area in the shortcut path, in which case the GH will send a message cascading back to the starting GH instructing it to abort the rerouting procedure. On receiving the REROUTE packet, the next hop temporarily caches the GIDs of the senders and the new next hop into a rerouting cache. Meanwhile, the next hop also invokes a timer for possible rollback. As this timer fires, the next hop will immediately update its routing table with this rerouting cache without having received the instruction to abort. Otherwise, the next hop will remove the rerouting cache and thereby cancel the modification of the routing table.

In the example in Fig. 5b, as the sink moves from Cell-1 to Cell-4, a new extended routing path (E, F, G, H) is exploited to attach the original routing path by the building routing method mentioned above. On receiving the data packet, the LGH in Cell-4 (node H) will examine the packet to find the potential new routing path (H, I, J, D) in the north discovering direction. Meanwhile, node H performs the rerouting procedure to build this new path as shown in Fig. 5c.

4 Simulation results

This section presents the results of the simulations that were conducted to compare the performance of EAGER to that of proposed grid-based protocol, EADA. This work was developed on J-Sim [15, 16], a Java-based network simulator. J-Sim is a component-based, compositional simulation environment similar to other network simulators such as ns-2 [19], SENSE [6], OMNeT++ [12, 22]. It provides an autonomous component architecture [18] that allows for the quick development of simulations by assembling an assortment of different components that exchange different types of message to communicate with one another. The designer is able to make use of JSim's existing component library, or alternatively new components can easily be customized through object-oriented programming.

The parameter settings in our simulation environment were as follows:

- Power consumption = 0.66 W, 0.359 W, and 0.035 W for transmitting, receiving, and idling, respectively
- Sink speed = each sink moved with a specified constant speed, following the random waypoint mobility model [4]
- Wireless transmission range for each node = 120 m
- Propagation of radio channels followed the free space model
- Total simulation duration = 120 seconds
- Source reported the same data to all sinks

The performance of EAGER is evaluated by comparing it with EADA in terms of three performance metrics, namely: total energy consumption, average delivery latency, and rerouting frequency. The total energy consumption is

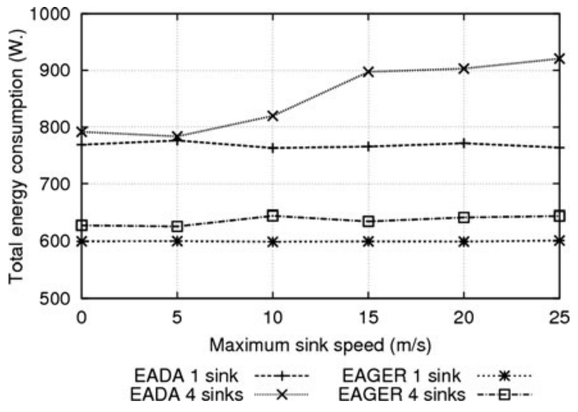

Fig. 6 Total energy consumption vs. maximum sink speed

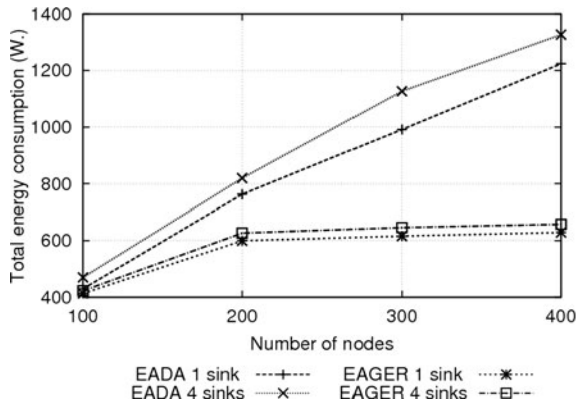

Fig. 7 Total energy consumption vs. number of nodes

defined as the communication energy (i.e., transmitting and receiving) and idling energy consumed by the network. The average delivery latency is defined as the average elapsed time between the moment a source transmits a packet and the time a sink receives the packet, thus indicating the overall speed with which data is reported from the source to the sink. Finally, the rerouting frequency is defined as the number of times that new dissemination paths were reconstructed during the total simulation period.

4.1 Performance analysis

The following subsections compare the performance of EAGER with that of EADA using different network sizes and sink speed.

4.1.1 Total energy consumption

This subsection studies the comparison of total energy consumption of the entire network. Figure 6 depicts the total consumed energy for different maximum sink speeds, which range from 0 to 25 m/s. Figure 7 shows the total consumed energy for the different numbers of sensor nodes, which range from 100 to 400.

Figure 6 displays the total consumed energy for different sink speeds. The simulation scenario assumes that there are 200 sensor nodes with one sink and four sinks, respectively, and that each sink moves at the same speed. As can be seen from Fig. 6, the total energy consumed by EAGER is less than that consumed by EADA. This is due to a number of reasons. First, EAGER uses a time-scheduling method that keeps some GHs sleeping—thus resulting in lower levels of energy consumption. Second, EAGER uses an approach that enables new queries to be resent in order to reconstruct new delivery paths whenever a mobile sink moves. EAGER caches the delivery path and evaluates whether it needs to reroute the new delivery path. This means that, in terms of handling sink mobility, EAGER does not perform rerouting as often as EADA. Also, EAGER's rerouting times are quicker than that of EADA. Therefore, the reconstruction of the delivery path requires significant energy use. As shown in Fig. 6, EADA needs to perform further reconstructing as sink speeds increase. Consequently, the total energy consumed by EAGER is far less than that consumed by EADA.

Figure 7 shows the total consumed energy for the different numbers of sensor nodes. The simulation scenario assumes that there is one sink and four sinks, respectively, and that each sink moves at a speed of 10 m/s. As shown in Fig. 7, the total energy consumed by EADA is more than that consumed by EAGER. This is due to a number of reasons. First, as the number of mobile sinks increases, EADA requires that all sinks perform frequent re-routing to change their delivery path. Second, re-routing at high node density causes more packet collisions and thus higher energy consumption. In contrast, EAGER assigns some GHs to sleep periodically to reduce energy dissipation. Figure 7 depicts how the total energy consumed by EADA increases linearly with the increase in node density, while the total energy consumed by EAGER does not increase with the increase in node density.

4.1.2 Rerouting overheads

This subsection compares the rerouting of overheads. Figure 8 shows the rerouting frequency for different maximum sink speeds, which range from 0 to 30 m/s. Figure 9 depicts the rerouting frequency for different numbers of nodes, which range from 100 to 400. Figure 8 shows the rerouting frequency for different maximum sink speeds. The simulation scenario assumes that there are 100 sensor nodes with one sink and eight sinks, respectively, and that every sink moves at the same speed. As is shown by Fig. 8, the rerouting frequency performed by EAGER is much less than that performed by EADA. The reason for this is that EAGER uses a rerouting approach to resend queries to reconstruct the new delivery paths. Unlike with EADA, EAGER does not need to reconstruct when a sink moves from one grid

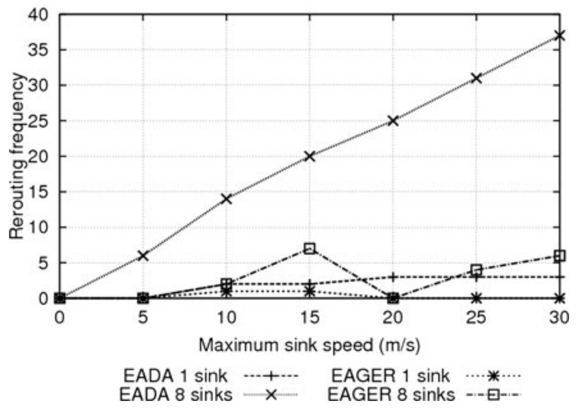

Fig. 8 Rerouting frequency vs. maximum sink speed

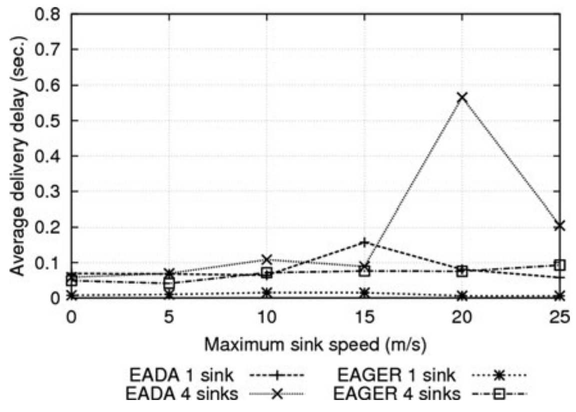

Fig. 10 Average delivery delay vs. maximum sink speed

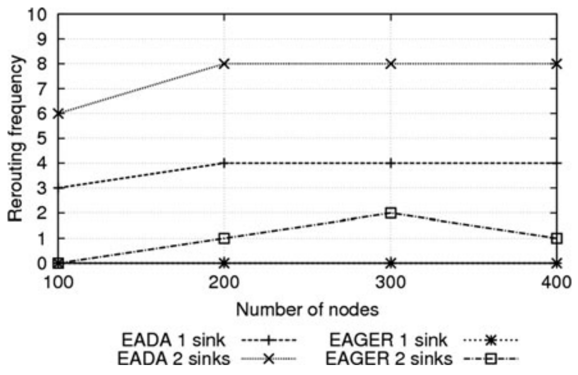

Fig. 9 Rerouting frequency vs. number of nodes

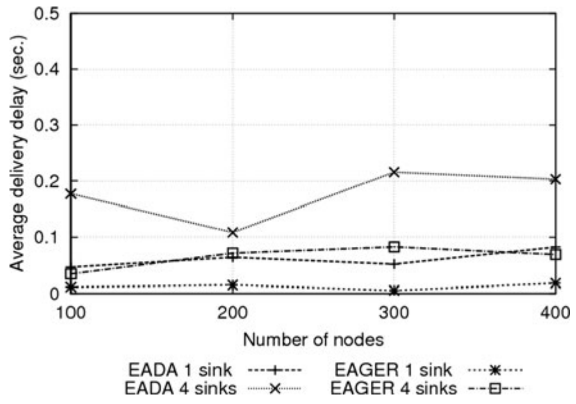

Fig. 11 Average delivery delay vs. number of nodes

cell to another. When the number of sinks equals one, the rerouting frequency of EAGER differs slightly from that of EADA. However, as the number of sinks increases, the frequency of EAGER is much lower than that of EADA.

Figure 9 depicts the rerouting frequency for different numbers of sensor nodes. The simulation scenario assumes that there is one sink and two sinks, respectively, and that each sink moves at a speed of 20 m/s. As shown in Fig. 9, the time of rerouting performed by EADA is more than that performed by EAGER. The reason for this is that, when the number of mobile sinks increases, the sinks in EADA will need to perform more rerouting to change their delivery path. The time for rerouting in EADA is proportional to the number of mobile sinks. In contrast, the time for rerouting in EAGER depends on the movement paths of sinks. As based on the aforementioned algorithm, EAGER does not perform rerouting when sinks move to grid cells that are further away. Therefore, the rerouting frequency of EAGER is significantly less than that of EADA.

4.1.3 Average delivery latency

This subsection compares the average delivery delays of EAGER and EADA. Figure 10 shows the average delivery delay for the different maximum sink speeds, which range from 0 to 25 m/s. Figure 11 shows the delivery delay for the different numbers of nodes, which range from 100 to 400.

In the example in Fig. 10, the simulation scenario assumes that there are 200 sensor nodes with one sink and four sinks, respectively, with every sink moving at the same speed. Figure 10 shows that the average delivery delay with EAGER is less than that of EADA. The reason for this is that EAGER uses a rerouting approach to reconstruct new delivery paths, thereby optimizing the new delivery paths and reducing delivery delay. When the number of sinks is four and each sink is moving over 15 m/s (as shown in Fig. 10), the average delivery delay achieved by EAGER is significantly shorter than that with EADA.

Figure 11 depicts the average delivery delay for different numbers of sensor nodes. The simulation scenario assumes that there are one sink and four sinks, respectively, and that each sink is moving at a speed of 10 m/s. Figure 11 shows how the average delivery delay achieved by EAGER is much shorter than that with EADA. There are a number of reasons for this. First, while EADA reconstructs delivery paths with limited flooding in order to handle sink mobility, it does not consider optimizing the new reconstructed paths. As EADA's grid gateways deliver data packets by such paths, delivery delays may increase. Alternatively, EADA's use of limited flooding to reconstruct the delivery paths can cause packet collisions, thereby extending the delivery delay. In contrast, EAGER applies a different approach to reconstruct and shorten delivery paths instead of using the EADA limited flooding method. As a result, EAGER performs far better than EADA in terms of delivery delay.

5 Conclusion and future work

As an important field of emerging technology, wireless sensor networks (WSN) offer many new possibilities for target tracking [20, 21, 28, 32] and environmental surveillance [17] by allowing the observer to move around freely. However, the dissemination of sensing data to the mobile observer still presents significant challenges from the viewpoint of the design of routing schemes [7]. Moreover, WSN applications are also constrained by limited energy resources, which necessarily affects the lifetime of the WSN. Much research has been conducted into the dissemination of protocols with regards to either achieving effective data dissemination or energy efficiency, while also working to satisfy the requirements of the mobile observer. Almost all of this research uses frequent rerouting as a means of resolving the issue of mobility. However, in doing so, rerouting operations lead to increased overheads and energy consumption, resulting in a trade-off between the need for rerouting to optimize the network operation and that of maximizing network lifetime. This paper has unveiled the novel EAGER routing protocol, which is able to prolong the lifetime of WSN through energy efficiency, while also improving the efficiency of data dissemination under multiple mobile sinks. In terms of saving energy, EAGER is based on the virtual-grid structure and keeps one node—the Grid Head (GH)—active within each grid cell to disseminate data. EAGER also applies a time-scheduling method to allow all idle GHs to sleep for set periods at specific intervals. The use of this method increases energy efficiency, while ensuring that the radio channels in any three of four adjacent cells are available at any time, though without creating any void communication channels. In terms of disseminating data under multiple mobile sinks, EAGER uses a rerouting approach to identify and reconstruct new data dissemination paths between multiple mobile sinks and the source. This approach enables the reduction of overheads related to rerouting frequency, as well as handling the issue of sink mobility. Finally, an extensive simulation was developed to allow for the comparison of EAGER with EADA, an alternative scheme that has been proposed. The simulation results show that EAGER is not only able to accommodate the challenges posed by mobile sinks, but is also able to conserve energy more effectively than EADA.

In this paper, some issues need to be addressed for the future work. As the mentioned assumption, all sensor nodes are homogeneous and synchronized at startup. Performing the time-scheduling of EAGER depends on this assumption. Once some nodes are heterogeneous instead, considering the synchronization issue is significantly important because incorrect time-slots period will affect the dissemination of data. However, network-wide synchronizing leads further overheads. Under the constrained-resource of WSNs, it also brings challenges.

References

1. Akyildiz, I., Su, W., Sankarasubramaniam, Y., & Cayirci, E. (2002). A survey on sensor networks. *IEEE Communications Magazine*, *40*(8), 102–114.
2. Al-Karaki, J. N., & Kamal, A. E. (2004). Routing techniques in wireless sensor networks: a survey. *IEEE Wireless Communications*, *11*(6), 6–28.
3. Albowicz, J., Chen, A., & Zhang, L. (2001). Recursive position estimation in sensor networks. In *Proceedings of network protocols ninth international conference* (pp. 35–41).
4. Broach, J., Maltz, D. A., Johnson, D. B., Hu, Y.-C., & Jetcheva, J. (1998). A performance comparison of multi-hop wireless ad hoc network routing protocols. In *Proceedings of the fourth annual ACM/IEEE international conference on mobile computing and networking* (pp. 85–97).
5. Chang, R.-S., & Lee, A.-C. (2007). An energy efficient data query architecture for large scale sensor networks. *IEICE Transactions on Communication*, *90-B*(2), 217–227.
6. Chen, G., et al. (2005). *SENSE—sensor network simulator and emulator*. http://www.ita.cs.rpi.edu/sense/index.html.
7. Jeon, H., Park, K., & Choo, H. (2009). Sink-oriented dynamic location service for shortest path relay with energy efficient global grid. In *Proceedings of ACM symposium on applied computing* (pp. 2174–2179).
8. Luo, H., Ye, F., Cheng, J., Lu, S., & Zhang, L. (2005). TTDD: two-tier data dissemination in large-scale wireless sensor networks. *Wireless Networks*, *11*(1–2), 161–175.
9. Mauve, M.H., Widmer, J., & Harenstein, H. (2001). A survey on position-based routing in mobile ad hoc networks. *IEEE Network Magazine*, *15*(6), 30–39.
10. Mir, Z.-H., & Ko, Y.-B. (2007). A quadtree-based hierarchical data dissemination for mobile sensor networks. *Telecommunication Systems*, *36*(1–3), 117–128.
11. Nakamura, E., Nakamura, F., Figueiredo, C. M., & Loureiro, A. (2005). Using information fusion to assist data dissemination in wireless sensor networks. *Telecommunication Systems*, *30*(1–3), 237–254.
12. *OMNeT++ discrete event simulation system*. http://www.omnetpp.org/.
13. Quy, N. X., Min, D., & Choi, E. (2008). An energy-efficient data dissemination protocol for grid-based wireless sensor networks. In *Proceedings of the IEEE international conference on research, innovation and vision for the future* (pp. 141–146).
14. Savvides, A., Han, C. C., & Srivastava, M. B. (2001). Dynamic fine-grained localization in ad-hoc networks for sensors. In *Proceedings of the international conference on mobile computing and networking* (pp. 166–179).

15. Sobeih, A., Chen, W.-P., Hou, J. C., Kung, L.-C., Li, N., Lim, H., Tyan, H.-Y., & Zhang, H. (2006). J-sim: a simulation and emulation environment for wireless sensor networks. *IEEE Wireless Communications*, *13*(4), 104–119.
16. Sobeih, A., Chen, W.-P., Hou, J. C., Kung, L.-C., Li, N., Lim, H., Tyan, H.-Y., & Zhang, H. (2005). J.-Sim: A Simulation and Emulation Environment for Wireless Sensor Networks. In *Proceedings of the 38th Annual Simulation Symposium* (pp. 175–187).
17. Solano, W. M., Junell, J., Schmalzel, J. L., & Shumard, K. C. (2004). Implementation of wireless and intelligent sensor technologies in the propulsion test environment. In *Proceedings of sensors for industry conference* ISA/IEEE Press, New York (pp. 135–138).
18. *The autonomous component architecture*. http://sites.google.com/site/jsimofficial/component-arch.
19. *The network simulator–ns-2*. http://www.isi.edu/nsnam/ns/.
20. Tsai, H.-W., Chu, C.-P., & Chen, T.-S. (2007). Mobile object tracking in wireless sensor networks. *Computer Communications*, *30*(8), 1811–1825.
21. Tseng, Y.-C., Kuo, S.-P., Lee, H.-W., & Huang, C.-F. (2004). Location tracking in a wireless sensor network by mobile agents and its data fusion strategies. *Computer Journal*, *47*(4), 448–460.
22. Varga, A. (1999). Using the OMNeT++ discrete event simulation system in education. *IEEE Transactions on Education*, *42*(4), 11.
23. Wang, H., Elson, J., Girod, L., Estrin, D., & Yao, K. (2003). Target classification and localization in habitat monitoring. In *Proceedings of the IEEE international conference on acoustics, speech, and signal* (pp. 6–10).
24. Wang, N.-C., Huang, Y.-F., Chen, J.-S., & Yeh, P.-C. (2007). Energy-aware data aggregation for grid-based wireless sensor networks with a mobile sink. *Wireless Personal Communications*. doi:10.1007/s11277-007-9325-9.
25. Wang, S.-T., & Wu, J.-L. C. (2004). SABAGAR: a simple attribute-based addressing and GPS-aided routing protocol for applications in wireless sensor networks. *Telecommunications Systems*, *26*(2–4), 197–212.
26. Xu, Y., Heidemann, J., & Estrin, D. (2001). Geography informed energy conservation for ad hoc routing. In *Proceedings of ACM mobile computing and networking* (pp. 70–84).
27. Xuan, H. L., Sed, D. H., & Lee, S. (2005). Minimum-energy data dissemination in coordination-based sensor networks. In *Proceedings of the IEEE 11th international conference on embedded and real-time computing systems and applications* (pp. 381–386).
28. Yang, H., & Sikdor, B. (2003). A protocol for tracking mobile targets using sensor network. In *Proceedings of the first IEEE international workshop on SensorNetwork protocols and applications* (pp. 71–81).
29. Yedavalli, K., & Krishnamachari, B. (2008). Sequence-based localization in wireless sensor networks. *IEEE Transactions on Mobile Computing*, *7*(1), 81–94.
30. Yick, J., Mukherjee, B., & Ghosal, D. (2008). Wireless sensor network survey. *Computer Networks*, *52*(12), 2292–2330.
31. Zhang, R., Zhao, H., & Labrador, M. A. (2006). The anchor location service (ALS) protocol for large-scale wireless sensor networks. In *Proceedings of the first international conference on integrated Internet ad hoc and sensor networks Article No. 18*.
32. Zhang, W., & Cao, G. (2004). DCTC: dynamic convoy tree-based collaboration for target tracking in sensor networks. *IEEE Transactions on Wireless Communications*, *3*(5), 1689–1701.
33. Zhao, H., Labrador, M. A., & Zhang, R. (2006). A grid-based sink location service for large-scale wireless sensor networks. In *Proceedings of international conference on wireless communications and mobile computing* (pp. 689–694).

Yuan-Po Chi received the College Diploma in Chemistry Engineering from National Taipei University of Technology in 1988, the M.S. degree in Computer Science and Information Engineering from National Cheng Kung University, Taiwan, in June 1997 and the Ph.D. degree in Computer Science and Engineering from National Chung-Hsing University, Taiwan, in August 2012. His current research interests include mobile computing, wireless networks, embedded systems and compiler techniques.

Hsung-Pin Chang received the B.S., M.S., and Ph. D degrees in Computer and Information Science from National Chiao Tung University, Taiwan. He was a postdoctoral researcher in National Chiao Tung University in 2002–2003, and an assistant professor in Electrical Engineering at National Changhua University of Education, Chaunghua, Taiwan, from 2003 to 2004. Since 2004, he has been with the Department of Computer Science and Engineering at National Chung Hsing University, Taiwan, where he is now an associate professor. His research interests include wireless sensor networks, network protocols, operating systems, and embedded systems.