Adaptive route-sharing protocol for data collection in Wireless Sensor Networks

Chih-Yung Chang* and Yun-Jung Lu

Department of Computer Science and Information Engineering,
Tamkang University,
Tamsui, Taipei 25137, Taiwan
E-mail: cychang@mail.tku.edu.tw
E-mail: yjlu@wireless.cs.tku.edu.tw
*Corresponding author

Jang-Ping Sheu

Department of Computer Science,
National Tsing Hua University,
Hsinchu 30013, Taiwan
E-mail: sheujp@cs.nthu.edu.tw

Chung-Yu Wang

Institute for Information Industry,
Taipei 10622, Taiwan
E-mail: fujima@nmi.iii.org.tw

Abstract: A Wireless Sensor Network (WSN) comprises a large number of sensor nodes and a few sink nodes. When multiple sink nodes are interested in collecting the readings of the same monitoring region, it is conducive to exploit the sharing route to save bandwidth and power consumption and prolong WSN’s lifetime. This paper proposes a Dynamic Route-Sharing Protocol (DRSP), which constructs sharing routes based on different attributes (for example, frequency, packet length or delay time) of the commands requested from different sink nodes. The proposed DRSP dynamically adjusts data transmission route to achieve the goals of routes sharing and route length reduction. The simulation study shows that DRSP saves more energy and bandwidth consumptions than the existing work and thus prolongs the WSN’s lifetime.

Keywords: WSN; wireless sensor network; data collection; routing; network lifetime.


Biographical notes: Chih-Yung Chang received the PhD degree in Computer Science and Information Engineering from National Central University, Taiwan, in 1995. He is currently a Full Professor of Department of Computer Science and Information Engineering at Tamkang University, Taiwan. He served as an Associate Guest Editor of Journal of Information Science and Engineering (JISE, 2009), Telecommunication Systems (TS 2010), IET Communications 2011 and a member of Editorial Board of International Journal of Ad Hoc and Ubiquitous Computing (JAHU C, 2009). His current research interests include vehicular ad hoc networks, wireless sensor networks, WiMax broadband networks and mobile computing.

Yun-Jung Lu received the BS in Computer Science from Alethia University, Taiwan, in 2006, and the MS from Computer Science and Information Engineering of Tamkang University, Taiwan, in 2008, where he is working towards his PhD. He won lots of scholarships in Taiwan and participated in many Wireless Sensor Networking projects. His research interests include IEEE 802.11 and mobile computing, ad hoc and sensor network and networking planning.

Jang-Ping Sheu received the BS degree in Computer Science from Tamkang University, Taiwan, Republic of China, in 1981, and the MS and PhD degrees in computer science from National Tsing Hua University, Taiwan, Republic of China, in 1983 and 1987, respectively. He is currently a Chair Professor of the Department of Computer Science, National Tsing Hua University. His current research interests include wireless sensor networks, vehicular ad hoc
1 Introduction

Wireless Sensor Networks (WSNs) are potentially applied in many fields such as military monitoring, location tracking, environmental sampling and healthcare monitoring. A WSN is mainly composed of few sink nodes and a large number of sensor nodes (Sausen et al., 2009). The sink nodes play the role of interface between users and WSNs, enabling users to dispatch the queries to a region of sensor nodes in a wireless manner. Each sensor node has sensing, computation and wireless communication capabilities. Deployed in the monitoring region, the sensor nodes are responsible for collecting environmental information and reporting their readings according to the queries sent from sink nodes (He et al., 2008). Since sensor nodes are powered by batteries, energy conservation is one of the most important issues in developing routing protocol.

Previous work (Al-Karaki and Al-Malkawi, 2008; Chen et al., 2002) indicated that communication among sensor nodes is the major source of energy consumption. To prolong the network lifetime of the WSN, designing an energy-efficient routing protocol that takes energy consumption into consideration has been a critical issue and has received much attention. Heinzelman et al. (1999) proposed an energy-efficient routing protocol, named LEACH, which uses cluster head to collect the readings from cluster members and then the head forwards the readings to the sink node. Instead of directly transmitting data from each sensor to sink node, the LEACH saves the significant energy consumption. However, a single-hop environment, which means all sensors neighbour to the sink node, is assumed. Furthermore, the energy consumption of LEACH is not efficient in case that cluster head and sink have a large distance.

To improve LEACH, a number of studies (Intanagonwiwat et al., 2003; Sausen et al., 2009; Ye et al., 2005) have proposed energy-efficient routing protocols for a multi-hop WSN. Previous researches (Intanagonwiwat et al., 2002; Kim et al., 2004; Mikami et al., 2006) emphasised the importance of building a shared route between multiple sink nodes and a data source node or between a sink node and multiple data source nodes. Route-sharing protocols were proposed to reduce the total length of shared route so that the energy consumption for data collection can be saved. However, the pre-constructed route cannot be adaptively changed according to the new request of sink nodes.

Kim et al. (2004) adopted a greedy algorithm to construct a shared route for collecting the same data from a specific region to different sink nodes. Then, the branch nodes in the tree filter the data to each route branch according to the frequencies defined by the sink node. Zeng et al. (1998) proposed a multicast tree topology, which is constructed for gathering data from a sensor, which is the root of the multicast tree. When a sink node intends to join the multicast tree, the sink will send the joining message to the root of the multicast tree, and the root of the multicast tree subsequently relays the joining message to the sensor node on the tree, which is nearest to the sink node. After that, the sensor node will calculate the gravity point to build a sharing route for the new sink node. However, the location of the gravity point might not be deployed any sensor, and hence the gravity point might not be the optimal branch point. Moreover, how to build a multicast tree after finding gravity point is not explicitly stated in the paper. Usually, building such a tree requires flooding operations to attain the goal. A large amount of control packets are thereby produced. On the branch point of each multicast tree, the same data must be periodically updated to support the request of each sink node. Thus, the branch point will encounter the same power-consumption problem existed in research (Bhattacharya et al., 2005), and previous built multicast tree will be destroyed.

This paper proposes a DRSP, which constructs routes based on data requests from multiple sink nodes. The established sharing routes can be dynamically adjusted according to newly built route under cost-efficient condition, improving the share degree among existing routes and hence prolonging the network lifetime.

The remaining sections in the paper are organised as follows: Section 2 introduces the related work and basic concepts of the proposed protocol. Section 3 proposes the considered network environment and problem statement while Section 4 details the proposed DRSP. Section 5 compares the performance of DRSP against related work, and finally, a brief conclusion is presented in Section 6.

2 Related work and basic concept

Data collection is the major task of WSNs. On constructing the routes for data collections, researches (Intanagonwiwat et al., 2003; Ye et al., 2005) considered requests sent from multiple sink nodes in different time intervals and aimed to provide data transmission service by establishing different
and direct communication routes. However, the route construction did not take the sharing path into consideration, causing energy and bandwidth consumptions. As shown in Figure 1(a), sink nodes $K_1$ and $K_2$ apply the routing protocols (Intanagonwiwat et al., 2003; Ye et al., 2005) and build a direct communication route to the coordinator independently. In total, there are seven sensor nodes, including $s_1, s_2, s_3, s_4, s_6, s_7$ and $s_9$, participating in the route.

Kim et al. (2004) present a route construction protocol, which considers path sharing to reduce the number of forwarding nodes in the route and thus saves the energy and bandwidth consumptions. As shown in Figure 1(b), the first route is constructed based on the request sent by sink node $K_1$. After that, another sink node $K_2$ sends a similar but different request to the coordinator $s_1$. The routing protocol tries to construct a route that can partially overlap with the existing routes. As a result, there are totally five sensors, including $s_1, s_2, s_3, s_4$ and $s_9$, involved in the routes from the coordinator $s_1$ to sinks $K_1$ and $K_2$. The constructed routes have higher degree of path sharing and save sensor nodes $s_4, s_6$ and $s_7$. The route sharing not only saves energy and bandwidth consumptions but also avoids the transmission of replicate data on the network, reducing the phenomenon of packet contention and collisions.

**Figure 1** The comparison of constructed routes from sink nodes $K_1, K_2$ and $K_3$ to coordinator $s_1$ using different approaches: (a) direct path; (b) two sink nodes sharing path; (c) three sink nodes sharing path and (d) dynamic sharing path (see online version for colours)

Though the sharing route (Kim et al., 2004) can reduce the number of forwarding nodes, however, the degree of route sharing can be further improved. Kim et al. presented the concept that the sink node sequentially connects to the existing routes. The later the sink node sends the data retrieval request to the coordinator, the more difficult for constructing an optimal sharing route since it has a constraint that the latter route should be attached to the existing routes. As shown in Figure 1(c), suppose that the sink nodes $K_1$ and $K_2$ have constructed two routes Path 1 and Path 2, respectively. When the sink node $K_3$ intends to periodically retrieve data from the same coordinator, the proposed protocol will further invite one more forwarding node $s_7$ to build the new sharing route. Thus, the sharing routes connecting the coordinator to the three sink nodes totally contain sensor nodes $s_2, s_3, s_5, s_7$ and $s_9$. However, the constraint that the latter route should be attached to the existing route can be further removed to seek to an optimal sharing route. As shown in Figure 1(d), whenever sink node $K_3$ intends to retrieve data from the coordinator, a new route can be constructed and the optimal route might be the one that changes the existing routes, enabling the old routes attaching to the new route. As a result, only sensor nodes $s_4, s_5, s_7$ and $s_9$ join the route. Compared with the route shown in Figure 1(c), the route constructed in Figure 1(d) saves three forwarding nodes.

The aforementioned example shows that the cost of sharing route and power dissipation cannot be efficiently saved by using greedy algorithm, which aims to attach a new route to the existing one. If the existing routes can be adequately adjusted when constructing a new route, the number of forwarding nodes might be further reduced and hence the energy and bandwidth consumptions for redundant packet forwarding can be saved. This paper proposes a protocol, named DRSP, which builds high degree of sharing routes and enables the data collection to be more energy efficient.

### 3 Network environment and problem statement

This section presents the network model and assumptions. Some notations that will be used in the proposed DRSP are given.

#### 3.1 Network environment and problem statement

The considered WSN is composed of a few sink nodes, a large number of sensor nodes and a few coordinators. All coordinators and sensor nodes are aware of their own location information. The sensor nodes are responsible for performing monitoring tasks including sensing the environmental information and periodically transmitting their readings to the coordinators according to the sinks’ requests. On the basis of user’s request, sink node will send command to all the sensor nodes within a specific region so as to meet the requirement of gathering data periodically. The coordinator takes the responsibility of data collection from sensor nodes within the specific region and then proceeds with the data calculation and reports results to the sink nodes. It is permitted that the times of sending requests by different sink nodes can be different. The content of request sent by sink node includes the returning frequency required for data collection, the attributes of the content of the collected data, the expected share degree of data collecting route, the permitted delay time of returning data and the time interval for data collection. Because of the need for returning data periodically, when a sink node sends
a request to a coordinator, the coordinator will transmit collected data from itself to the sink node through the constructed route in a multi-hop manner. In the literature, many routing algorithms have been proposed to cope with the problem of how coordinator gathers data from sensor nodes within a specific region in an energy saving and efficient way. This paper would not discuss the issue of gathering data from sensor nodes to the coordinator. Alternatively, we discussed how to build efficient routes with high sharing degree from the coordinator to multiple sink nodes according to their requests.

### 3.2 Notations

To clearly present the details of the proposed protocol, the following notations are defined. Assume that there are $k$ sink nodes $K$ and $m$ sensor nodes $s_j$ in a WSN, where $1 \leq i \leq k$ and $1 \leq j \leq m$. Notation $R$ denotes the coordinator. The command requested from each sink node $K_i$ contains the following possible attributes. Notation $f_i$ denotes the required query-frequency and notation $a_y$ denotes the attributes of the content of collected data. The expected share degree of routes for data collection is denoted by $l_i$. The permitted delay time for each queried data is denoted by $d_i$. Notation $T_i$ represents the time interval for data collection. Consider the process that the sink node $K_i$ sends Route Request ($RREQ$) packets to the coordinator $R$ using directional flooding approach. If node $s_x$ sends an $RREQ$ packet to node $s_y$, we refer node $s_x$ to be the upstream candidate of node $s_y$. A route constructed from sink $K$ to coordinator $R$ is denoted by Route $(K, R) = \{K = s_0, s_1, \ldots, s_y = R\}$ where $s_i$ denotes the nodes that participate in the route. We further assume that node $s_i$ is the upstream node of node $s_i + 1$, and node $s_i + 1$ is the downstream node of node $s_i$. Table 1 summarises the notations used in this paper.

<table>
<thead>
<tr>
<th>Table 1: Notations used in this paper</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k$</td>
</tr>
<tr>
<td>$K_i$</td>
</tr>
<tr>
<td>$m$</td>
</tr>
<tr>
<td>$s_j$</td>
</tr>
<tr>
<td>$R$</td>
</tr>
<tr>
<td>$f_i$</td>
</tr>
<tr>
<td>$a_y$</td>
</tr>
<tr>
<td>$l_i$</td>
</tr>
<tr>
<td>$d_i$</td>
</tr>
<tr>
<td>$T_i$</td>
</tr>
</tbody>
</table>

The so-called existing route denotes a built route from some sink nodes to the coordinator while the current route denotes an under-constructed route from the new sink node to the coordinator. Let current node denote the sensor node that is on the current route and executes the route construction task.

Let the existing forwarding node denote the sensor node that is on the existing route. Moreover, to construct a low-cost sharing route, notation $C_{ij}$ is defined as the cost of link $s_i \rightarrow s_j$. The permitted delay time for each queried data is denoted by $d_i$. Notation $T_i$ represents the time interval for data collection. Consider the process that the sink node $K_i$ sends Route Request ($RREQ$) packets to the coordinator $R$ using directional flooding approach. If node $s_x$ sends an $RREQ$ packet to node $s_y$, we refer node $s_x$ to be the upstream candidate of node $s_y$. A route constructed from sink $K$ to coordinator $R$ is denoted by Route $(K, R) = \{K = s_0, s_1, \ldots, s_y = R\}$ where $s_i$ denotes the nodes that participate in the route. We further assume that node $s_i$ is the upstream node of node $s_i + 1$, and node $s_i + 1$ is the downstream node of node $s_i$. Table 1 summarises the notations used in this paper.

The so-called existing route denotes a built route from some sink nodes to the coordinator while the current route denotes an under-constructed route from the new sink node to the coordinator. Let current node denote the sensor node that is on the current route and executes the route construction task.

Let the existing forwarding node denote the sensor node that is on the existing route. Moreover, to construct a low-cost sharing route, notation $C_{ij}$ is defined as the cost of link $s_i \rightarrow s_j$, which is measured by the cost of power consumption and time delay. Let $C_{i \rightarrow j}$ denote the cost of a route from sensor node $s_i$ to node $s_j$, we have

$$C_{i \rightarrow j} = \sum_{k=1}^{j} C_{s_{k+1} s_k}.$$

As shown in Figure 2, we define the terms of the cost for each route. The current route cost $C_{K_i \rightarrow s_j \rightarrow R}$ denotes the cost for building route between sink node $K_i$ and coordinator $R$. On the current route, the cost of the route from sink node $K_i$ to current node $s_j$ is referred as the front cost of current route or the front cost in short, and is denoted by the symbol $C_{K_i \rightarrow s_j}$. Furthermore, the cost of route between current node $s_j$ and existing forwarding node $s_j$ is referred as branch route cost, and is denoted by $C_{s_j \rightarrow s_j}$.

**Figure 2** The illustration of the cost of different routes (see online version for colours)

### 4 Dynamic Route-Sharing Protocol

This paper focuses on the route construction between multiple sinks and one coordinator and proposes a DRSP. The proposed DRSP protocol is composed of three major Phases: Route Request, Route Reply and Route-Sharing Phases. In the Route Request Phase, a sink node sends user’s commands to coordinator by applying directional flooding operations in a multi-hop manner. Then, an initial route will be built between the sink node and the coordinator in the Route Reply Phase. After that, the Route-Sharing Phase will dynamically adjust the constructed route to establish an optimal sharing route according to the cost calculation. The following describes the details of DRSP.
4.1 Route request phase

When any user issues a command to a certain sink node $K_i$, the sink node $K_i$ will initiate the Route Request Phase. In this phase, the sink node $K_i$ sends $RREQ$ to the coordinator $R$ using directional flooding approach in a multi-hop manner. Figure 3 depicts the format of $RREQ$ packet, which contains the query request of data collection parameters sent from sink node. An $RREQ$ packet is composed of several fields including sink ID and coordinator ID, frequency, attributes, share degree, delay time and time interval. In addition, the $RREQ$ packet also contains the information of upstream candidate including its ID, remaining energy and the front cost of sender.

![Image](image.png)

**Figure 3** The packet format of the $RREQ$

<table>
<thead>
<tr>
<th>Sink_ID</th>
<th>Coordinator_ID</th>
<th>$F_{\text{req}}$</th>
<th>Attribute</th>
<th>Share degree</th>
<th>Delay time</th>
<th>Time interval</th>
<th>Sender_ID</th>
<th>$E_{\text{rem}}$</th>
<th>$C_{\text{sink}}$</th>
</tr>
</thead>
</table>

To relay the $RREQ$ packets to the coordinator $R$ as soon as possible but avoid the packet flooding over the entire sensor network, DRSP will allow the $RREQ$ packet to be flooded within a Request Zone, which is defined according to the locations of sink node $K_i$ and the coordinator. Without the loss of generality, assume that the sink node is located at the right bottom of the coordinator. The Request Zone can be defined as the region where the top left and the bottom right points are the coordinates of coordinator and sink node, respectively. Let $C_{\text{lowest}}$ denote the lowest cost from a sink node to a sensor node. Since the sensor in the Request Zone might receive multiple $RREQ$ packets and therefore has different front costs, it should record $C_{\text{lowest}}$ in its own cost table. Figure 4(a) gives an example to illustrate the operation of Route Request Phase. The sink node $K_i$ appoints a certain Request Zone to forward $RREQ$ packets to coordinator $R$. Assume that node $s_3$ is located in the Request Zone and first receives the $RREQ$ packet from its neighbouring node $s_2$. In the meantime, sensor node $s_3$ will treat node $s_2$ as an upstream candidate and evaluate the costs $C_s^+_{s,2}$ and $C_{s,2}^-$. On the basis of the content of $RREQ$, a new value of the front cost $C_{s,3}^- = C_{s,3}^- + C_s^+$ is calculated. Since sensor node $s_3$ first receives the $RREQ$ packet, it sets $C_{\text{lowest}}$ value to be the front cost $C_{s,3}^-$. In its own cost table, places $C_{s,3}^-$ in the fields of $RREQ$ packet and records the upstream node to be node $s_2$. After that, node $s_3$ sends the $RREQ$ packet to its neighbours. Whenever sensor node $s_3$ receives another $RREQ$ packet from node $s_5$, similarly, sensor node $s_3$ will treat node $s_5$ as its upstream candidate, and calculate the front cost $C_{s,5}^-$. If node $s_3$ finds that the cost value $C_{s,5}^- = C_{s,5}^- + C_s^-$ of the route passing through node $s_5$ is better than the cost $C_{s,3}^- = C_{s,3}^- + C_s^+$ of the route passing through only node $s_5$, node $s_3$ will set $C_{\text{lowest}}$ to be the value of $C_{s,5}^-$. update the upstream node to be node $s_5$, put the value $C_{s,5}^-$ in the $RREQ$ packet and then send the $RREQ$ packet again.

![Image](image.png)

**Figure 4** The steps of building routes between sink node $K_i$ and coordinator $R$: (a) in Request Zone, sink node $K_i$ uses directional flooding approach to send the $RREQ$ packet to the coordinator $R$, (b) Coordinator $R$ responds $RREP$ to the sink node $K_i$ according to the cost table and (c) optimistic node sends $S-RREQ$ packet to construct the sharing route (see online version for colours)

4.2 Route reply phase

In this phase, coordinator $R$ will determine a route between sink node $K_i$ and coordinator $R$, and then sends back a Route Reply Packet (RREP) to sink node $K_i$ in a multi-hop manner. Moreover, the coordinator will notify sink node $K_i$ whether this is the first route. If it is the case, the route is determined by coordinator $R$. Otherwise, a share route will be further explored. The format of $RREP$ is shown in Figure 5. The $\text{Receiver_ID}$ field records the upstream node whereas the $\text{Destination_ID}$ field records the sink node that sent the request. In this way, the $RREP$ can be transmitted to the sink node in a multi-hop manner. Moreover, the First route field records whether there has already existed a route. The $\text{C_{sink-toR}}$, $\text{C_{sink-toS}}$ and $\text{C_{next-hop}}$ fields record the current route cost, cost from a sink to a sensor node and branch route cost, respectively, to help determine whether it is worth to construct a share route. The $\text{Hop_{max}}$ field records the degree of route sharing whereas the $\text{F_{current}}$ field records the frequency of data collection requested by the sink node.

![Image](image.png)

**Figure 5** The packet format of the $RREP$

<table>
<thead>
<tr>
<th>Sender_ID</th>
<th>Receiver_ID</th>
<th>Destination_ID</th>
<th>First route</th>
<th>$C_{\text{sink}}$</th>
<th>$C_{\text{sink}}$</th>
<th>$C_{\text{next-hop}}$</th>
<th>$\text{Hop_{max}}$</th>
<th>$\text{F_{current}}$</th>
</tr>
</thead>
</table>

Figure 4(b) is taken as an example to illustrate the detailed operations of Route Reply Phase. Herein, we assume that sink node $K_i$ is the first sink node that issues command to the coordinator $R$. The coordinator $R$ has two neighbours of...
upstream sensor nodes $s_6$ and $s_7$ at the moment. We further assume that coordinator $R$ can derive the minimal current route cost $C_{k_i \rightarrow r}$ by constructing a route passing through node $s_5$. Thus, coordinator $R$ will fill the values of $C_{k_i \rightarrow r}$ in the two fields of $RREP$: current route cost and the front cost, and set the first route field in $RREP$ to 1. Then, coordinator $R$ sends the $RREP$ to sink node $K_i$ in a multi-hop manner.

Figure 4(c) illustrates the operations of the other situation that sink node $K_i$ is not the first sink node but issues the command to coordinator. Since there are existing routes to coordinator $R$, the first route field in the $RREP$ is filled in 0 by coordinator $R$, which informs sink node $K_i$ to exploit the opportunity for building sharing routes and notifies the current node, which is responsible for forwarding $RREP$, to construct sharing routes in its surroundings. Here, we define the $l_i$-hop neighbours of the current node as the optimistic node, where value $l_i$ denotes the sharing degrees decided by the sink node. The 1-hop neighbours of coordinator $R$ and current nodes also receive $RREP$. The optimistic nodes will treat $RREP$ as $S$-$RREQ$ packet since the first route field in $RREP$ is set to 0.

### 4.3 Route-sharing phase

After entering the Route-Sharing Phase, the optimistic nodes will send the $S$-$RREQ$ packet to find outward sharing routes, and the hop distance of sending $S$-$RREQ$ packet outward has followed the value of the share degree $l_i$ field in $RREP$. Therefore, all the sensor nodes within the requested hop count of $S$-$RREQ$ packet will play the role of optimistic node. A forwarding node receiving an $S$-$RREQ$ packet represents the current and existing routes are able to construct a sharing route. Consider two paths $P_1$ and $P_2$ as shown in Figure 6(a) where $K_i$ and $K_j$ are sink nodes and $R$ is a coordinator. Without loss of generality, we assume that it is worth to change path of $P_1$ from $\{K_i, A, R\}$ to $\{K_j, A, B, R\}$. That is, we assume that the following expression is satisfied so that it is worth to change path $P_1$.

As a result, paths $P_1$ and $P_2$ have sharing segment for saving the redundant data transmissions.

$$C_{K_i \rightarrow A} + C_{A \rightarrow B} < C_{K_j \rightarrow R}.$$  

Figure 6(b) depicts the sharing path of $P_1$ and $P_2$, where $\{B, R\}$ is the shared segment. To describe the detailed operations of DRSP, the following defines branch node and attached node. A branch node of a shared path is referred to the node that is the first common node of the shared segment. The attached node of path $P_1$ is the first node that changes path $P_1$ so that $P_1$ can be attached to path $P_2$ and therefore the two paths can be merged to construct a common segment. As shown in Figure 6(b), node $B$ is a branch node while node $A$ is an attached node.

In the Route-Sharing Phase, optimistic node will try to find an adequate couple of branch and attached nodes to change the route for achieving the goal of route sharing. Since the requested returning frequency of sink node is not the same as that of coordinator, the following route-changing strategy is given as a foundation for building a sharing route.

![Figure 6](image)

**Figure 6** The illustration of constructing a sharing path: (a) assume that it is worth to change path from $P_1 = \{K_i, A, R\}$ to $\{K_j, A, B, R\}$ and (b) the branch and attached nodes are nodes $B$ and $A$, respectively (see online version for colours)

**Criterion 1:** When building a sharing route, the route with lower requested returning frequency must be attached to those routes with higher requested returning frequencies.

When the two routes with different returning frequencies intend to retrieve data from the same coordinator at the same time, the route with higher returning frequency can share similar data with the route with lower returning frequency. Criterion 1 helps reduce the cost of data transmission.

**Criterion 2:** If optimistic node $s_i$ satisfies the following expression (3), it can continue to send $S$-$RREQ$ packet. The forwarding node $s_i$ on the existing route whose cost calculation meets expression (3) returns the request of changing route from the current nodes.

$$C_{K_j \rightarrow s_i} + C_{s_i \rightarrow s_j} < C_{K_j \rightarrow s_j}.$$  

In Figure 7, it can be found that after node $s_{13}$ delivers the $RREP$ to its upstream node $s_{10}$, optimistic node $s_{14}$ will treat the $RREP$ as $S$-$RREQ$ packet, and check if the cost of the route satisfies the requirements as stated in Criterion 2. If $C_{K_j \rightarrow s_{10}} + C_{s_{10} \rightarrow s_{13}} < C_{K_j \rightarrow s_{13}}$, node $s_{14}$ will stop to send the $S$-$RREQ$ packet. On the contrary, if criterion 2 is satisfied, node $s_{14}$ will update the value of branch route cost in $S$-$RREQ$, and continue to send $S$-$RREQ$ packet to its neighbouring nodes $s_{14}$ and $s_{15}$, aiming to construct the share route. Similarly, nodes $s_7$ and $s_{15}$ will decide whether to continue to send $S$-$RREQ$ packet.

As shown in Figure 7, the $S$-$RREQ$ packet is sent to two nodes $s_2$ and $s_3$ on the existing route. Then, nodes $s_2$ and $s_3$ evaluate whether the cost of route meet the requirement of route adjustment based on Criterion 2, play the role of branch node candidate according to Criterion 1, and then reply to the current node $s_{13}$ with an $S$-$ACK$ message. At this moment, the current node $s_{13}$ is referred as attached node candidate, and branch node candidates $s_2$ and $s_3$ will add cost into the fields of $S$-$ACK$ as a reference for optimistic nodes and branch node candidates to further construct the sharing routes. Figure 8 shows the packet format of $S$-$ACK$. The **Branch_node_candidate_ID** and **Attached_node**
_candidate_ID fields record the IDs of branch and attached node candidates. The C_sink→candidate_ID fields record the cost of segment from sink node to branch node candidate. The F_existing node records the frequency of the existing route so that the current sink node can figure out the frequency of the existing route. The Join field helps to record whether the current route should be connected to the existing route. If it is the case, the value of Join is zero. Otherwise, its value is one.

**Figure 7** Sink node $K_2$ uses RREP and S-RREQ packet to create possible sharing route (see online version for colours)

As shown in Figure 7, current node $s_{18}$ sends S-RREQ packet to the optimistic node $s_7$ via optimistic node $s_{14}$. Before deciding to send S-RREQ packet, node $s_7$ will check the S-RREQ packet and find that the remaining hop is 1. Node $s_7$ then checks the value $C_{s_7,s_{14}}$ and evaluates that the cost of route passing through the optimistic node $s_{14}$ is 22.2 whereas the cost of route passing through the optimistic node $s_{15}$ is 17.6. Thus, the requirement of Criterion 2 is satisfied. The optimistic node $s_7$ will further continue to forward the S-RREQ packet. When current node $s_{19}$ sends S-RREQ packet to the optimistic node $s_7$ via optimistic node $s_{15}$, it will find that the remaining hop count of S-RREQ packet is 0. Thus, optimistic node $s_7$ will only keep the information in the cost table instead of continuing forwarding S-RREQ packet.

The behaviours of sending S-RREQ packet and S-ACK by the current node and optimistic node might identify multiple couples of attached and branch node candidates. Thus, to decide an optimal couple of attached and branch nodes, the current node, which plays the attached node candidate, will send an S-ACK packet to the sink node so that the sink node can select the sharing route with the lowest cost and notify the optimal couple of attached and branch nodes and the coordinator to construct the connections.

As shown in Figure 9, upon receiving S-ACK sent from branch node candidate $s_3$, the current node $s_{13}$ will send the S-ACK to sink node $K_2$.

**4.4 Adaptive route sharing**

Previously, we have mentioned the strategy of building sharing route when current route has sensor nodes with low returning frequency. The following states the case when the returning frequency of current route is higher than that of the existing route.

**Figure 9** An example of selecting a branch node candidate (see online version for colours)

Different from the aforementioned strategy, upon receiving the S-RREQ packet, the existing forwarding node, say $s_x$, will evaluate the necessity of building sharing route between current route and existing route based on the route cost from sink node to both itself and the branch route cost. If it is the case, node $s_x$ will fill the Join field in S-ACK with value 1, representing that the existing route will join current route for the sake of forming sharing route. Then, node $s_x$ will play the role of attached node candidate and send the S-ACK packet to the current node. Upon receiving the S-ACK message, the current node will play the role of branch node candidate and simultaneously forward the S-ACK message to the sink node.

Finally, the sink node can determine the best attached and branch nodes from all possible candidates and then notify them and the coordinator to construct the connections. As a result, the existing route with lower frequency can attach to the current route with higher frequency for constructing a more efficient sharing route.

As depicted in Figure 10, suppose the existing forwarding node $s_7$ receives an S-RREQ packet from the current node $s_{12}$. It is not worth for node $s_7$ to change the route according to cost evaluation. Upon receiving the S-RREQ packet sent from current node $s_{20}$, the existing forwarding node $s_{13}$ decides to build a sharing route with node $s_{20}$ and then play the role of attached node candidate. After that, node $s_{13}$ returns an S-ACK message to node $s_{20}$ to join current route. Node $s_{20}$ then sends the S-ACK to the sink node $K_2$. Upon receiving the S-RREQ packet, the existing forwarding node $s_3$ helps combine original existing route and the current route as a new sharing route and the original existing route can reduce the cost by changing to a new sharing route. The node $s_2$ then returns S-ACK to node $s_{12}$.  

<table>
<thead>
<tr>
<th>Branch node candidate</th>
<th>Attached node candidate</th>
<th>$C_{sink→}$</th>
<th>$F_{existing}$</th>
<th>Join</th>
</tr>
</thead>
</table>

Each current node sends S-RREQ packet independently to find possible branch node candidates. For an optimistic node, it might receive different S-RREQ packets from different current nodes. As shown in Figure 7, current node $s_{18}$ sends S-RREQ packet to the optimistic node $s_7$ via optimistic node $s_{14}$. Before deciding to send S-RREQ packet, node $s_7$ will check the S-RREQ packet and find that the remaining hop is 1. Node $s_7$ then checks the value $C_{s_7,s_{14}}$ and evaluates that the cost of route passing through the optimistic node $s_{14}$ is 22.2 whereas the cost of route passing through the optimistic node $s_{15}$ is 17.6. Thus, the requirement of Criterion 2 is satisfied. The optimistic node $s_7$ will further continue to forward the S-RREQ packet. When current node $s_{19}$ sends S-RREQ packet to the optimistic node $s_7$ via optimistic node $s_{15}$, it will find that the remaining hop count of S-RREQ packet is 0. Thus, optimistic node $s_7$ will only keep the information in the cost table instead of continuing forwarding S-RREQ packet.

The behaviours of sending S-RREQ packet and S-ACK by the current node and optimistic node might identify multiple couples of attached and branch node candidates. Thus, to decide an optimal couple of attached and branch nodes, the current node, which plays the attached node candidate, will send an S-ACK packet to the sink node so that the sink node can select the sharing route with the lowest cost and notify the optimal couple of attached and branch nodes and the coordinator to construct the connections.

As shown in Figure 9, upon receiving S-ACK sent from branch node candidate $s_3$, the current node $s_{13}$ will send the S-ACK to sink node $K_2$.

**4.4 Adaptive route sharing**

Previously, we have mentioned the strategy of building sharing route when current route has sensor nodes with low returning frequency. The following states the case when the returning frequency of current route is higher than that of the existing route.

**Figure 9** An example of selecting a branch node candidate (see online version for colours)

Different from the aforementioned strategy, upon receiving the S-RREQ packet, the existing forwarding node, say $s_x$, will evaluate the necessity of building sharing route between current route and existing route based on the route cost from sink node to both itself and the branch route cost. If it is the case, node $s_x$ will fill the Join field in S-ACK with value 1, representing that the existing route will join current route for the sake of forming sharing route. Then, node $s_x$ will play the role of attached node candidate and send the S-ACK packet to the current node. Upon receiving the S-ACK message, the current node will play the role of branch node candidate and simultaneously forward the S-ACK message to the sink node.

Finally, the sink node can determine the best attached and branch nodes from all possible candidates and then notify them and the coordinator to construct the connections. As a result, the existing route with lower frequency can attach to the current route with higher frequency for constructing a more efficient sharing route.

As depicted in Figure 10, suppose the existing forwarding node $s_7$ receives an S-RREQ packet from the current node $s_{12}$. It is not worth for node $s_7$ to change the route according to cost evaluation. Upon receiving the S-RREQ packet sent from current node $s_{20}$, the existing forwarding node $s_{13}$ decides to build a sharing route with node $s_{20}$ and then play the role of attached node candidate. After that, node $s_{13}$ returns an S-ACK message to node $s_{20}$ to join current route. Node $s_{20}$ then sends the S-ACK to the sink node $K_2$. Upon receiving the S-RREQ packet, the existing forwarding node $s_3$ helps combine original existing route and the current route as a new sharing route and the original existing route can reduce the cost by changing to a new sharing route. The node $s_2$ then returns S-ACK to node $s_{12}$.
and performs the route-changing scheme to achieve the goal of route sharing.

The proposed protocol tries to build a sharing route for those sink nodes that request to collect the data from the same coordinator. For the data with different attributes, another route to coordinator \( R \) will be built by sink node itself as shown in Figure 11. We assume that the coordinator \( R \) chooses nodes \( s_{10}, s_{11} \), and \( s_{34} \) to forward data from the coordinator \( R \) to the sink node \( K_s \). However, the common attributes of the requested data by \( K_4, K_1, K_2 \), and \( K_3 \) are \((a_{41}, a_{42})\) and there is still a different attribute \((a_{44})\). After receiving responded \( S-\text{ACK} \) from \( s_{12} \), node \( s_{34} \) can know the attributes \((a_{41}, a_{42})\) of shared data, which is offered by node \( s_{12} \).

Figure 10 The existing route formed by sink node \( K_1 \) and \( K_2 \), and the illustration of steps in building sharing route among sink node \( K_3 \) (see online version for colours)

Figure 11 The sharing route that sink nodes \( K_1, K_2, K_3 \) and \( K_4 \) builds to connect coordinator \( R \) (see online version for colours)

4.5 Discussions of cost function design

In the section, Cost Function is proposed for each sensor node to calculate the cost of communication with neighbouring nodes at the moment when sensor node receives packets, which is sent from neighbouring nodes. For sink node \( K_i \) the request of its query is frequency \( f_i \). Assume that there are \( \delta \) attributes \( a_{ij} \) in the query, where \( 1 \leq j \leq \delta \). If a route built by sink node \( K_i \) is \( P = \{ K_i, s_{01}, s_{11}, ..., s_{m-1}, R \} \), the distance between \( s_k \) and \( s_{k-1} \) is \( r_{sk} \) and the residual power of sensor node \( s_k \) is \( 0 \leq v_k \leq 1 \). The cost of route \( P \) can be denoted by the following expression:

\[
\text{Cost}(P) = \sum_{k=0}^{m} a_i \times \frac{1}{v_k} \times r_{sk}^2
\]  

(4)

The number of neighbours contending to send data will be the main factor of the delay of packet transmission. Thus, the delay of route \( P \) can be denoted by the following expression:

\[
\text{Delay}(P) = \sum_{k=1}^{m} \text{Delay} \left( S_j \right) = \sum_{k=1}^{m} \sum_{x \in \text{route}} x \cdot \text{route}
\]

(5)

From equations (4) and (5), the Cost Function route \( P \) is:

\[
C_P = \alpha \times \text{Cost} (P) + (1 - \alpha) \times \text{Delay} (P)
\]  

(6)

4.6 The DRSP algorithm

This subsection formally presents the DRSP algorithm.

The Dynamic Route Sharing Protocol (DRSP)

Phase I: The Route Request Phase:

Step 1: The sink creates an \( \text{RREQ}(\text{Sink ID}, \text{Coordinator ID}, \text{Frequency}, \text{Attributes}, \text{Sharing degree}, \text{Delay time}, \text{Sender ID}, \text{Current_Route}) \) packet and simply broadcasts it.

Step 2: Upon receiving the \( \text{RREQ} \) packet from upstream candidate \( s \), node \( s' \) evaluates the front cost \( C_{\text{sink}, \text{sink}} = C_{\text{sink}, \text{sink}} \). Node \( s' \) records upstream node \( s \), sets \( \text{RREQ}, C_{\text{sink}, \text{sink}} = C_{\text{sink}, \text{sink}} \) and broadcasts the \( \text{RREQ} \) packet.

Phase II: The Route Reply Phase:

Step 1: The coordinator selects the route with lowest cost, creates an \( \text{RREP}(\text{Sender ID}, \text{Receiver ID}, \text{Destination ID}, \text{First route}, \text{Cost}, \text{Hopcount}, \text{Correlation}) \) packet and broadcasts it.

Step 2: Upon receiving the \( \text{RREP} \) packet, the current and optimistic nodes check whether or not \( \text{RREP}. \text{first route} = \text{true} \). If it is the case, current node broadcasts the \( \text{RREP} \) packet. Otherwise, the optimistic node will treat \( \text{RREP} \) packet as \( \text{S-RREQ} \) packet and then goto the Route Sharing Phase.

Phase III: The Route Sharing Phase

Step 1: Upon receiving the \( \text{S-RREQ} \) packet from sensor node \( p \), optimistic node \( q \) evaluates the cost from the sink to itself and branch costs by
5 Simulations

This section evaluates the efficiency of the proposed DRSP. The proposed DRSP mechanism was implemented in GloMoSim (Zeng et al., 1998) and was compared with two mechanisms: SAFE (Kim et al., 2004) and Flooding. In the simulation, the sink nodes and sensor nodes are randomly deployed in the monitoring region with size $2000 \times 2000$ m$^2$. The number of sensor nodes is varied ranging from 300 to 500 and the number of sink nodes varies ranging from 1 to 10. A sensor node is chosen to play the role of coordinator, which is responsible to collect and transmit data to multiple sink nodes according to each sink node’s query. Sensor nodes and sink nodes know their own location, and all sink nodes are also aware of the location of the coordinator. The common communication range of sink node and sensor node is 50 m. The energy consumption of sensor node for transmitting and receiving a packet is 1 and 0.5 µJ/bit (Kim et al., 2004), respectively. Table 2 summarises the parameters set in the simulation.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simulator</td>
<td>GloMoSim</td>
</tr>
<tr>
<td>Node deployment</td>
<td>Random</td>
</tr>
<tr>
<td>Monitor Region</td>
<td>$2000 \times 2000$ m$^2$</td>
</tr>
<tr>
<td>The number of sensor node</td>
<td>300–500</td>
</tr>
<tr>
<td>The number of sink node</td>
<td>1–10</td>
</tr>
<tr>
<td>Location information</td>
<td>Known</td>
</tr>
<tr>
<td>The communication range of sensor node</td>
<td>50 m</td>
</tr>
<tr>
<td>The energy consumption of transmitting</td>
<td>1 µJ/bit</td>
</tr>
<tr>
<td>The energy consumption of receiving</td>
<td>0.5 µJ/bit</td>
</tr>
<tr>
<td>The size of query packet</td>
<td>36 bytes</td>
</tr>
<tr>
<td>The size of data packet</td>
<td>64 bytes</td>
</tr>
</tbody>
</table>

Each sink node will separately construct the route with the coordinator in order. The coordinator will periodically transmit collected data based on the data attributes and returning frequency of requested from sink node. The sizes of query and data packets are set at 36 and 64 bytes, respectively. Each result is obtained from an average of 10 experiments. The 95% confidence interval is always smaller than $\pm 5\%$ of the reported values.

Figure 12 discusses the relationship between the number of sink nodes and the average number of sharing route. The number of deployed sensors is set at 500. In general, the average numbers of sharing routes constructed by DRSP and SAFE increase with the number of sink nodes. The DRSP and SAFE have similar performance and significantly outperform the Flooding mechanism in terms of the average number of shared routes. For instance, when the numbers of sink node in the network are 5 and 10, DRSP constructs 4.21 and 8.678 sharing routes, respectively, while SAFE constructs 4.35 and 8.783 sharing routes, respectively. The main reason that SAFE constructs more shared routes than DRSP is that SAFE uses flooding approach to build sharing route. Thus, SAFE exploits more opportunities for finding the sharing route than DRSP. The Flooding scheme builds route from each sink node to the coordinator by applying flooding operations, but the construction of sharing route is never considered. Therefore, the possibility of generating sharing route increases with the number of sink nodes.

Figures 13 and 14 investigate the control overheads and the number of forwarding nodes of the compared mechanisms, respectively. There are totally three sink nodes and 500 sensor nodes deployed in the monitoring region. Figure 13 compares the number of control packets of the three compared mechanisms with share degree ranging from 0 to 7. The numbers of control packets of SAFE and Flooding mechanisms keep a constant even though the requirements for share degree are changed. This is because that both SAFE and Flooding mechanisms perform flooding...
operations over the network, regardless of the requirement of share degree. The number of control packets created by the proposed DRSP increases with the requested share degree. When the value of the required share degree is larger than 5, the number of control packets created by DRSP would not change drastically. The main reason is that all sensors in the network help to broadcast the control packets in case that the required share degree is larger than 5.

Figure 12 The relationship between the number of sink nodes and the average number of sharing routes (see online version for colours)

Figure 13 The relationship of share degree and control packet (see online version for colours)

Figure 14 The relationship between share degree and the number of forwarding nodes on sharing route (see online version for colours)

Moreover, two factors, including average energy consumption and control overhead, are also considered. In average energy consumption, the average power dissipation in route construction and data dissemination are mainly considered. In control overhead, the necessary control packet, which is issued in route construction, is considered. The dissipated power of each sink node and coordinator is shown as Figure 16. It can be found that DRSP dissipates less power than SAFE and Flooding. The main reason is that DRSP uses flooding approaches in local area, reducing the redundant control packets flooded over the network. Figure 17 depicts that DRSP consumes less power than SAFE and Flooding. The main reason is that when DRSP builds sharing route, it will adjust sharing route to the optimal sharing route based on the requested data attributes and returning frequency of sink node. Therefore, DRSP consumes less power than SAFE, which applies greedy algorithm to build route.

Figure 16 The power dissipation in route construction (see online version for colours)
Figure 17 The power dissipation in data dissemination (see online version for colours)

Figure 18 shows the overhead of building route between sink node and coordinator. It can be easily found that DRSP uses local flooding approach to find routes and its control overhead is roughly 3020 when there are 10 sink nodes in the network. But, the control overhead of SAFE and Flooding are 21785 and 48289, respectively.

Figure 18 The relationship between the number of sink node and control overhead (see online version for colours)

6 Conclusion

In the paper, a protocol that builds sharing route between multiple sink nodes and coordinator is proposed. The construction of sharing route can significantly reduce the number of forwarding nodes, the bandwidth waste and transmission of replicate data, achieving the goal of power saving. When the sink node requests data similar to the other sink nodes, the proposed DRSP tries to create a sharing route, and dynamically adjust the route according to requested attribute and returning frequency. Simulation results reveal that DRSP outperforms the existing schemes SAFE and Flooding and saves considerable energy consumptions.

References


Bibliography


