TARS: An Energy-Efficient Routing Scheme for Wireless Sensor Networks with Mobile Sinks and Targets

Yuan-Po Chi  
Department of Computer Science and Engineering  
National Chung-Hsing University  
Taichung, Taiwan  
phd9404@cs.nchu.edu.tw

Hsung-Pin Chang  
Department of Computer Science and Engineering  
National Chung-Hsing University  
Taichung, Taiwan  
hpchang@cs.nchu.edu.tw

Abstract—Wireless sensor networks (WSNs), which are operated with strict resources, offer promise for surveillance technology including target tracking and environmental monitoring. In some sensor network applications, aggregating and delivering sensing data from mobile targets to mobile sinks is necessary. Nevertheless, such mobility can cause the established routes non-optimal and even broken, and require repeated route recovery procedures. To address this issue, previous TRENs routing protocol presented a tracking-assisted routing scheme to avoid the need for frequent rerouting. In this paper, we propose an enhancement of TRENs, called TARS, that also deals with the routing issue for WSNs consisting of multiple mobile targets and sinks. Compared with TRENs, TARS proposes an enhanced tracking technology and a lightweight shortcutting approach to decrease redundant messages, so as to increase the efficiency of routing procedures and to optimize the routing paths in the context of dynamic topology. Besides, TARS introduces a timescheduling method to control the radio channels of idle nodes to save more energy. Comparing TARS with previously proposed grid-based schemes through extensive simulations, we unveil that our scheme conserves more energy and achieves better performance in aspect of specified evaluation metrics.

Keywords: wireless sensor networks, virtual-grid structure, routing, mobile sinks, mobile targets, energy-efficient

I. INTRODUCTION AND MOTIVATION

A wireless sensor network (WSN), consisting of enormous sensor nodes with constrained resources, is an emerging technology for tracking object and observing environment. Each sensor node has sensing devices, being thus capable of sensing, measuring and collecting environmental data [1]. In addition, each sensor node is equipped with wireless networking interfaces to communicate with each other on an ad hoc basis. All sensor nodes are pre-deployed in a monitored area. After deployment, once one or more sensor nodes sense the target stimulus in their monitored area, one node (known as the "source") immediately reports back the relevant data, such as location and temperature, to the observer (known as the "sink") via the wireless channel to a laptop or hand-held mobile device [2]. In this paper, sinks and observers are used interchangeably.

Previous researches on routing algorithms in wireless sensor networks usually focus on stationary sink and targets. Nevertheless, in numerous applications, observing and tracking multiple moveable targets, such as wild animals and enemy vehicles, by mobile observers are required. For instance, biologists can drive a car to collect and study the behavior of wildlife; the soldier deploys sensor nodes over a battlefield and drives a jeep to monitor the movements of the enemy’s tanks. As the scenarios of those examples, both observers and targets can move anywhere at any time. As a result, network topology would change so frequently that the established routes soon become non-optimal and even broken. To repair these obsolete routes, excessive topology maintenance and frequent rerouting are required and can cause significant control and management overhead. However, sensor nodes are usually battery-powered and thus are energy-limited. Consequently, disseminating data efficiently while coping with the sink and target mobility is a key design consideration of the routing protocols in WSNs.

To address these issues, a plethora of routing and dissemination methods have been proposed previously. Some schemes proposed to use the virtual-grid structure for efficient data dissemination while reducing the energy consumption [3-8]. For example, based on the virtual-grid structure, Xuan and Lee presented a data dissemination model, called CODE [5] to establish data dissemination paths between sources and mobile sinks. Furthermore, CODE adopts the agent approach for WSNs with mobile multi-sinks environment. Alternatively, EADA enhances CODE to reduce frequency of rerouting to saving energy [6]. EADA establishes the delivering paths between sources and mobile sinks with fan-like flooding area for saving energy. However, during the establishing period, the overlapping fan-like area caused by multiple mobile sinks may consume more energy.

Previously, we also presented a virtual-grid-based routing scheme, named TRENs, in mobile sinks and targets environment [16]. TRENs utilized both a tracking-assisted method to deal with the dynamic topology and a shortcutting method to optimize the delivering paths. However, each time when delivering sensing data to the sink node, TRENs needs to record all visited forwarding nodes in the packet to evaluate whether a possible shortcut is existed or not. As a result, the
more number of the visited nodes, the more increased size of the data packet, which in turn consumes more energy during wireless transmission [17]. Therefore, in this paper, we propose a new approach, entitled Trace-Announcing Routing Scheme (TARS) to enhance TREN. TARS is also based on the virtual grid structure. But TARS is more energy-efficient than TREN in aspect of handling mobility and shortcutting. First, TARS eliminate the redundant messages used in TREN to track mobile sinks and targets. Besides, TARS adopts an improved shortcutting method to decrease the number of visited node in TREN, so that TARS reduces the length of every forwarded mobile sinks and targets. Therefore, in this paper, we propose TREN and other grid-based protocols through the extensive simulation, this paper unveils that TARS conserves more energy and achieves better performance.

The rest of this paper is organized as the followings. Section II describes related work. Section III presents the architecture and algorithm of TARS. Section IV results the performance evaluation of TARS. Finally, the conclusion is given in Section V.

II. RELATED WORK

Many grid-based routing protocols have been proposed for WSNs [3-8, 16]. For example, CODE [5] relies on the virtual-grid structure and revises the GAF protocol [18] to establish data dissemination paths between sources and mobile sinks. CODE selects a coordinator in each grid cell as a dissemination node. When detecting the target’s stimulus, the source floods a message containing its location to inform all coordinators before reporting data. Using this informed location, the sink then builds a routing path, toward the source by forwarding a data query message. Moreover, CODE utilizes an agent approach to deal with the mobility of sink. Once the sink moves, it polls nearby coordinators for choosing a new agent. The new agent is responsible for rebuilding the new routing path and notifying the old agent to remove the obsolete routing path. However, those rebuilding operations entail explicit overhead and have a significant overhead. If the sink moves quickly, the new agent must perform the rebuilding process frequently, which incurs a significant overhead. In addition, while the source reports data during the rebuilding period, the data delivered along the existed obsolete routing path is possibly lost.

The same as 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 [6]. In addition, EADA confines the forwarding area of query messages sent by the sink within a fan-as zone to avoid broadcast-storm. Using such a confined zone, EADA also limits the forwarding number of query messages in order to eliminate rerouting overhead while handling sink mobility. However, if multiple mobile sinks move within the monitored area, this rerouting approach with overlapped fan-as zones may lead to additional communication overheads. All coordinators within the overlap area need to relay lots of messages, meaning that they consume energy quickly. If some mobile sinks move quickly, the energy consumption of the affected coordinators could greatly affect the overall network lifetime.

Besides, there are many proposed protocols focus on the tracking and detouring techniques [9,13-15]. Zhang et al., proposed the concept of dynamic convoy tree-based collaboration scheme to detect and track a mobile target [9]. Datta et al., proposed a shortcut procedure to reduce the hop count of routing paths in wireless networks [14]. Similarly, Ma et al., unveiled a path pruning strategy to reduce the excessive number of hops caused by the detouring mode of geographical routing protocols [15].

III. ARCHITECTURE AND ALGORITHM

This section introduces the architecture and algorithm of TARS. There are some assumptions made as follows. First, each node is location-aware by equipping with GPS locator or other location estimation techniques. Second, all sensor nodes are homogenous with a synchronized built-in time clock. Third, all sensor nodes are stationary and capable of sensing stimulus generated from the targets via the sensor channel. If multiple sensor nodes sense the target’s stimulus at same time, the node that receives maximum signal strength of stimulus becomes the source and generates data report to all sinks via the wireless channel. Each sink is capable of collecting target’s information from monitored area any time.

A. Construction of the Virtual-grid Structure

At the beginning, the monitored area is divided into several virtual grid cells. Each grid cell is associated with a unique pair of numbers, called grid identification (GID). Assume that the geographic position of origin in the monitored area is indicated by $(x_o, y_o)$. Furthermore, as shown in Figure 1, the grid size, $\alpha$, which is determined by the transmission range, $R_{tr}$, is defined as $R_{tr} / 2\sqrt{2}$. The sensing range, $S_{sn}$, is defined as $R_{tr}/2$. Under those ranges, every node of a grid cell can communicate with every node of its eight adjacent grid cells by the radio channel and sense targets within its grid cell by the sensing channel. Each sensor node calculates the GID of its belonging grid with its geographic coordinate $(x, y)$ by Eq. (1), where $\lceil k \rceil$ is largest integer less than $k$. According to Eq. (1), all deployed sensor nodes belonging to the same grid cell have an identical GID number.

$$GID = \left\lfloor \frac{x - x_o}{\alpha} \right\rfloor \quad \left\lfloor \frac{y - y_o}{\alpha} \right\rfloor$$

$\alpha :$ grid size

![Figure 1. Initializing Grid](image)
Then, all sensor nodes of each grid cell must elect one coordinator, called grid head (GH), for delivering data and performing routing and dissemination. The election policy is shown as following. Each node delays a random back-off time and then broadcasts an election packet with its GID. If the node makes an election attempt before receives any election packets from other grid members, this node becomes the grid head. Instead, if the node receives the election packet before its back-off time expired, its attempt is canceled immediately. Once the grid head is elected, all other grid members turn off their radio channels and only keep the sensor channel active to reduce the energy consumption. Finally, after the election of GH, each GH maintains a neighboring GH table by broadcasting a hello packet to all neighboring GHs. Once all GHs have cached the GIDs of all their neighboring GHs, the construction of the virtual grid structure is completed.

B. Time-Scheduling Radio Off Method

After the construction of the virtual grid structure, all of the elected grid heads are responsible for disseminating the sensing data. However, during the operation of sensor networks, not all of the grid heads will be participated in data dissemination. As a result, idle grid heads that always keep their radio channels active would only consume energy and cause a reduction in the lifetime of sensor networks. To tackle this problem, TARS utilizes a time-scheduling method that allows a number of the idle grid heads to be asleep, so as to allow these grid heads to turn off their radio to reduce the energy consumption.

The detailed mechanics are as follows. First, once a node has been elected as the GH, based on the sum of the x- and y-coordinates of its GID, it determines whether to keep its radio on or off. If the sum is even, the GH keeps its radio on. Otherwise, the GH turns off its radio with a fixed timer. Besides, it must inform all of its neighboring GHs about the timer value via the hello packet sent during the construction of the virtual grid structure. Consequently, the GH with its radio on would obtain the values of timers of all its four neighboring “radio-off” GHs. Then, these GHs turn off their radios immediately to save energy.

However, these radio-off GHs must turn on their radio periodically. As shown above, when a GH turns off its radio, it must also start a timer. When the timer fires, the GH turns on its radio to listen for incoming packets. Consequently, if an active, i.e., “radio-on”, GH wants to send packets to its neighboring inactive, i.e., “radio-off”, GHs, since it knows the timer values, it thus wants until the inactive GHs turning on their radios again. If no packet arrives, the inactive GH turns off its radio again.

C. Initial Routing Scheme

Before disseminating data, TARS needs to build the initial paths from sources to sinks by the routing operation. At first, when a node sense the target’s signal, it turns on its radio channel, becomes the source, and reports data back to the GH, called local GH (LGH), of its belonging grid cell. Similarly, if the radio of its LGH is off, the LGH would also be triggered to turn on its radio due to the sensing of the target’s signal. Notably, each radio-off GHs only turn off the radio but keep the sensor active. Thus, it can continue to sense the existence of target’s stimulus. Then, the LGH checks whether an existing routing path to the sink is existed in its routing information table (RIT). If the path is existed, the source reports the data to the sink by the path shown in the RIT. Otherwise, it floods an initial route request packet (IREQ) to find the path to the sink. For reducing energy consumption, only GH is responsible to receive and forward IREQ packet. The IREQ packet contains two fields: hop count and GH list. Initially, the value of hop count is zero and the GH list is empty. When a GH forwards the IREQ packet, it increases the value of hop count by 1 and adds its GID to the GH list. When the sink’s GH receives the IREQ packets, it chooses the IREQ packet whose hop count value is the smallest among all of the IREQ packets. Then, the sink’s GH sends back an initial route reply packet (IREP) to the source’s GH using the reverse visited path of the IREQ. In addition, intermediate GHs forwarding the IREP also build their RIT table simultaneously. Each entry of RIT is identified by a tuple of (destination, next, previous), where destination is the GID of the destination GH; next is the next hop from which the node routes the data; previous is the previous hop from which the node receives the data. Once the IREP reaches the GH of source node, the initial routing path is completely established. Thus, the source can disseminate the sensing data to the sinks via this initial routing path. All GHs along the initial routing paths are responsible for forwarding the data.

As shown in Figure 2, the LGH of the source, node A, issues the initial routing operation to build the initial routing path, (A, B, C, D, E, F), to the LGH of the sink, node F.

D. Trace-Announcing Routing Scheme

Since either sinks or targets may move anywhere at any time, the built initial routing paths might soon become broken. One possible solution is to re-construct the routing path each time when routing errors occur. However, re-constructing is energy consuming since it must flood repeated over the entire network. Thus, we propose a trace-announcing routing scheme to resolve this issue. Comparing with TRENz, TARS utilizes an enhanced mechanism to reduce redundant messages in aspect of handling the issue.

In the new proposed scheme, TARS tracks the moving path of sinks and targets passively. For easy of discussion, both sinks and targets are referred to as mobile objects (MO). When a MO appears, announcing the existence of MO is necessary. By means of the announcing information, TARS can solve the routing issue of mobile targets and sinks. The detailed procedures are as follows:
1. For mobile sinks: If the MO is the sink, the sink directly broadcasts an announcing packet to notify its LGH to start tracking its moving path via the radio channel. However, if the radio of its LGH is off, since the grid size is equal to \( R_x/2 \), thus the adjacent active GHs can also hear the announcing packet. Those adjacent GHs then relay the announcing packet to the “radio-off” LGH, when the LGH turns on its radio again. Besides, after receiving the announcing packet, the original “radio-off” LGH keep awake and does not turn off the radio. Upon moving, sink checks its current location by computing its GID. If moving to a new grid cell, the sink then notifies the new LGH and all of its neighboring GHs to track its successive moving path by broadcasting the announcing packet again.

2. For mobile targets: In contrast, if the MO is the target, since targets are unlikely to be equipped with wireless devices, the source must take responsibility for notifying the LGH of the source to capture the moving path of the MO. When one of the targets appears, the source broadcasts the announcing packet to notify its LGH immediately to start capturing the target’s moving path. Meanwhile, if the radio of the LGH is off, it can be also triggered to turn on its radio due to the sensing of the target’s signal. Besides, all of its neighboring GHs can capture the target’s successive moving path by receiving the announcing packet.

![Figure 3. Tracking information table for a mobile sink](image)

The **TRACE-ANNOUNCING** message simple contains the MO’s identifier and the GID of current location. Due to the assumed grid size, which is less than the communication range, all adjacent GHs can receive the **TRACE-ANNOUNCING** packet. Upon receiving the packet, each GH updates immediately its tracking information table (TIT) with the MO’s location information from the received packet. Each TIT entry is a tuple of \((MO_id, next, previous, GH_id)\). Each field of the tuple is defined as follows: 
- **next** is the location of the upstream hop for delivering the source’s reported sensing data;
- **previous** is the GID of the downstream hop;
- **GH_id** denotes the location of neighboring GH that caught the last track of the MO.

The upstream hop field and the downstream hop field indicate which grid cell that the MO has left and entered, respectively. By means of the TIT table, TARS can easily capture the moving paths of all mobile MOs and correct each initial routing path toward each MO’s current location. For example, as illustrated in Figure 3, when the MO, **Target1**, moves from grid A through grid B to grid C, each grid head maintains its TIT table from received **TRACE-ANNOUNCING** packet.

Consequently, each GH may maintain the RIT and TIT table simultaneously and thus it is possible that the RIT and TIT table in some GHs have the routing and tracking information for the same destination at the same time. Compared with the routing information, tracking information is considered new. Thus, TARS adopts the Using Trace Information First (UTIF) policy. In other words, TARS would search the TIT table first and, if the routing path does not exist in the TIT, then search the RIT table. For example, as shown in Figure 2, at beginning, the source reports the data by the initial routing path, (A, B, C, D, E, F). After a while, the sink moves from grid (0,6) to grid (1,4). Node D caches the moving path of the sink and creates a new entry in its TIT table for the sink. When node D wants to forward sensing data to the sink, it rules the UTIF policy to select the new routing path that is shown in gray arrow.

### E. Shortcutting Scheme

Since the sinks and targets move randomly, the forwarding path from the source to the sink may become curved. Previously, TRENs introduces a shortcutting scheme to optimize the routing and tracking paths. In TRENs, each data packet maintains two fields: hop count and visited GH list. The hop count is initialized to zero and increased each time when the data packet is forwarded by a GH. The visited GH list contains all of the GHs that forwards the data packet. Each entry of the visited GH list is organized by a tuple of \((GID, deliveryHopCount)\), where \(GID\) is a visited GH’s GID and \(deliveryHopCount\) denotes the delivery hop count from the LGH of the source to this visited GH. When each GH forwards the data packet, it increases the packet’s delivery hop count by one and records this count and its GID into the visited GH list.

As shown above, while delivering a data packet, TRENs needs to record the GIDs of all visited GHs into the visited GH list in the data packet to evaluate whether a possible shortcut is existed. If the packet is delivered through a long forwarding path, the number of the visited GH nodes may increase, which in turn leads to an increased packet size. It has been shown that the energy consumption of the wireless transmission is proportional to the length of packets. Besides, delivering a larger data packet in WSNs would increase the possibility of collisions, which in turn consume much more energy [17]. Consequently, TARS proposed an enhanced shortcutting method to reduce the size of the visited GH list for saving energy consumption. The same as TRENs, the shortcutting scheme of TARS also includes two steps: evaluation and shortcutting. As delivering the data packet each time, the immediate GH needs to perform the evaluation step for possible shortcuts. If a possible short is existed, the GH then performs the shortcutting step.

In the evaluation step, when delivering a data packet, TARS just only needs to record the number of delivering hop count and the GIDs of two kinds of GH nodes, including a key node (KN) and a curved node (CN), to thevisited list of the packet. A KN denotes a GH that is the end point of a shortcut.
When delivering a data packet, each forwarding GH checks whether the grid distance, which is called the grid hop number (GHN) and is calculated by Eq. (2), between itself and each entry of the visited list of the data packet is greater than the number of delivering hop count or not. If the GHN is greater than the delivering hop count, one possible shortcut is existed from the forwarding GH to the selected entry of the list. Then, the immediate GH removes all entries of the visited list and initializes this list with its GID as the KN node. Meanwhile, the GH performs the shortcutting step with the selected entry as the destination of a shortcut, i.e., KN node.

\[
\text{GHN}(A, B) = \sqrt{[\text{GID}(x(A) - \text{GID}(x(B))]^2 + [\text{GID}(y(A) - \text{GID}(y(B))]^2 \tag{2}
\]

In the shortcutting step, the forwarding GH that finds a possible shortcut sends a shortcutting packet to the shortcut’s destination. This packet contains the GIDs of the shortcut’s two endpoints: the destination is the ending point and the packet sender as the starting point. Once receiving the shortcutting packet, each immediate GH first determine the next hop for relaying by the algorithm \text{Shortcutting\_Nexthop} illustrated in Table I. Using the algorithm, each immediate GH searches one of neighboring GHs, which is closest to the direction of destination, from eight possible adjacent directions. Besides, if the next hop LGH is currently inactive, the sending of packets is hold until the next hop turns on its radio again. Notably, the absence of the next hop indicates a void area exists on this shortcut path, in which case the GH sends a message cascading backward to the starting point to abort this shortcutting procedure. When the next hop receives the shortcutting packet, it first computes the new next hop for relaying. Meanwhile, it caches temporarily the GIDs of the senders and the new next hop into a shortcutting cache. For possible rollback, it also invokes a timer. As receiving the abort message before this timer expires, it removes the shortcutting cache to cancel the modification of the routing table. Otherwise, it immediately updates its routing table with this cache.

**Figure 4.** Direction of Moving

**Figure 5.** Shortcut operation

<table>
<thead>
<tr>
<th>Table I.</th>
<th>Pseudo-Code of Finding Next Hop for Shortcutting</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Shortcutting_Nexthop</strong> (GH, ShortcuttingPacket dp)</td>
<td></td>
</tr>
<tr>
<td><strong>begin</strong></td>
<td></td>
</tr>
<tr>
<td>// dp: the Shortcutting Packet</td>
<td></td>
</tr>
<tr>
<td>// dp.EP: the ending point of Shortcutting Packet</td>
<td></td>
</tr>
<tr>
<td>if GH is equal to dp.EndingPoint then</td>
<td></td>
</tr>
<tr>
<td>stop</td>
<td></td>
</tr>
<tr>
<td>return nil</td>
<td></td>
</tr>
<tr>
<td>else</td>
<td></td>
</tr>
<tr>
<td>X.offset = dp.GID(x) – GH.GID(x)</td>
<td></td>
</tr>
<tr>
<td>Y.offset = dp.GID(y) – GH.GID(y)</td>
<td></td>
</tr>
<tr>
<td>Direction(x) = ( \begin{cases} 0 &amp; \text{if } X.offset = 0 \ \text{or } X.offset &lt; 0 \ X.offset/</td>
<td>X.offset</td>
</tr>
<tr>
<td>Direction(y) = ( \begin{cases} 0 &amp; \text{if } Y.offset = 0 \ \text{or } Y.offset &lt; 0 \ Y.offset/</td>
<td>Y.offset</td>
</tr>
<tr>
<td>Nexthop.GID(x, y) = GH.GID(x, y) + Direction(x, y)</td>
<td></td>
</tr>
<tr>
<td>if found Nexthop in GH’s neighbor list then</td>
<td></td>
</tr>
<tr>
<td>return Nexthop.ID</td>
<td></td>
</tr>
<tr>
<td>else</td>
<td></td>
</tr>
<tr>
<td>return nil</td>
<td></td>
</tr>
<tr>
<td><strong>end</strong></td>
<td></td>
</tr>
</tbody>
</table>

Figure 5 illustrates an example for the shortcutting scheme in details. The delivery path (A, B, C, D, E, F, G), becomes curved when the sink moves from grid cell GID(5, 0) to grid cell GID(5, 2). When delivering a data packet, each forwarding GH checks the upstream delivering direction and the downstream delivering direction. If the upstream direction is different from the downstream one, the forwarding GH appends its GID into the visited list of the packet before delivering. At beginning, node A first records itself as KN to the visited GH list of the data packet. In Figure 5, nodes B, C, E and G have different directions between upstream and downstream and thus are CNs. Therefore, their GIDs are inserted into the visited list. Upon receiving the data packet, node D finds the number of delivery hop count is three and is greater than the GHN between the node A and itself. Node D then resets the visited list with its GID and sends a shortcutting packet with the GID of node A as the destination of the next hop, i.e., node L, by the \text{Shortcutting\_Nexthop} algorithm to find a shortcut. After receiving the shortcutting packet, node L forwards the packet to the next hop and so no until the shortcutting packet reaches the destination node. Once the destination of the shortcut path, i.e., node A, receives this packet, the shortcut (D, L, A), which is shown with a gray-dotted arrow, is found. Following the same procedure, node H also finds another shortcut, (H, J, K, D). As a result, the original delivery path is replaced with the new shorter new, (A, L, D, K, H).
IV. PERFORMANCE EVALUATION

This section presents the performance evaluation of TARS. We developed a simulation based on J-Sim [10][11][12], a Java-based network simulator, to evaluate and compare TARS to other protocols, including EADA and TRENS. The parameters of power consumption are 0.66 W, 0.359 W and 0.035 W for transmitting, receiving and idling, respectively. There are 300 sensor nodes randomly distributed in $500 \times 500m^2$ field. Each sink and target moves following the random waypoint mobility model. The MAC layer used the IEEE 802.11 components of J-Sim [19]. The wireless transmission range of each node is 120 m. The duration of simulation is 120 seconds.

We use three metrics to evaluate TARS: total energy consumption, delivery ratio, and the average delivery latency. The total energy consumption is defined as the communication energy, includes transmitting and receiving, and the idle energy consumed during the simulation. The delivery ratio is the ratio of the number of received data packet at sink nodes to the total number of data packets generated by source nodes. This metric describes how effectiveness of the data delivery is. Finally, 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. This metric indicates how quickly the data is reported from the source to the sink.

A. Performance of Shortcutting

This subsection compares the performance of the shortcutting scheme of TARS with that of TRENS. This simulation includes two mobile sinks and four mobile targets. The speed assumptions are as followings: the speed of each sink ranges from 10m/s to 30m/s with an increment of 5m/s; the speed of each target varies 20m/s and 30m/s.

First, we present the shortcutting frequency under different moving speeds of the sinks and targets. The simulation result is shown in Figure 6. Obviously, the shortcutting time of TARS is far less than that of TRENS. Consequently, the shortcutting scheme proposed by TARS can reduce the shortcutting frequency, which in turn reduces the cost of shortcutting.

Alternatively, showing the total energy consumption of TARS and TRENS. Figure 7 results that the energy consumption of TARS is less than that of TRENS. The reason is that TARS uses the time-scheduling method to control the radios of grid heads and proposes the new shortcutting scheme to optimize the shortcutting performance to reduce the cost of shortcutting.

Figure 8 illustrates the delivery ratios of TARS, which keep above 80%, are better than that of TRENS. This is because, due to the proposed radio scheduling method, TARS reduces the possibility of collision while delivering packets. Besides, the result shows that TARS gains high delivery ratio than TRENS under the condition of high mobility.

Finally, Figure 9 shows that the average delivery latency of TARS is always better than that of TRENS, no matter what the
speed of sinks and targets are. It results that TARS uses the enhanced shortcutting scheme to find a better delivering paths. According to the simulation results, TARS can indeed minimize the cost of shortcutting, reduce the packet delivery latency, and maintain a high delivery ratio under the dynamic topology caused by mobile sinks and targets.

B. Comparison

Finally, this section presents the performance comparison between TARS and other virtual-grid-based protocols, including EADA and TRENS, in terms of the three metrics mentioned above. This simulation is developed with the assumption that there is one mobile target and four mobile sinks. Target moves with a constant speed at 5m/s, and the speed of sink ranges from 1 to 25m/s in an increment of 5 m/s.

Firstly, Figure 10 illustrated the total energy consumption. EADA is a grid-based protocol and improve the GAF scheme [18] to reduce the idle energy consumption. GAF just keeps all non-coordinator nodes to sleep periodically. In contrast, TARS and TRENS take a different approach that turn off the wireless devices of most sensor nodes to save energy. However, TARS improves the trace-announcing routing and the shortcutting scheme of TRENS to reduce the energy consumption further. Besides, TARS uses a radio scheduling method for grid heads to reduce the idle energy consumption. The simulation results, as plotted in Figure 10, show that TARS can indeed saves more energy than other protocols.

![Figure 10. Comparison of total energy consumption](image)

Next, Figure 11 shows the comparison for the average delivery ratio. To accommodate sink mobility, EADA uses an on-demand approach to determine when the coordinator node rebuilds the new delivery paths. If the sink moves far away from its original location, EADA rebuild the new delivery paths. Otherwise, the sink polls the neighboring gateways to search the original delivery paths instead of rebuilding paths. As the moving speed of sink is increased, the rebuilding action is increased that cause a decreased delivery ratio. In contrast, the tracking scheme used in TARS and TRENS can immediately catch the movements of all mobile sinks, allowing for the successful delivery of data packets to the sinks. In addition, TARS improves the shortcutting scheme of TRENS to optimize the delivery paths and thus obtain a higher delivery ratio. Overall, the delivery ratio of TARS is almost higher than that of all other methods under different moving speeds.

![Figure 11. Comparison of delivery ratio](image)

Finally, we show the comparison for the average delivery latency. EADA rebuilds the new delivery paths within fan-like zones when the sink moves far away the original delivery paths. The elapsed time of rebuilding of EADA affects the delivery latency. The more the number of sink is, the more the used fan-like zone is. Thus, it results in more delivery latency. In contrast, TARS utilizes the enhanced shortcutting scheme to optimize the delivery paths to reduce the delivery latency. Consequently, as shown in Figure 12, no matter the speed of the sink, the latency of TARS is less than that of EADA and TRENS.

![Figure 12. Comparison of average delivery delay](image)

V. CONCLUSION

Previously, we proposed a virtual-grid-based routing protocol, called TRENS, to deal with the context of dynamic topology for reducing power dissipation and improving routing performance. However, each time when delivering data, TRENS attaches the location information of all visited nodes into the data packet for evaluating whether a possible shortcut is existed. Such attached information would increase the size of the data packet, which in turn cause an increase of energy consumption for wireless transmission. Therefore, we propose a new approach, called Trace-Announcing Routing Scheme (TARS) to improve TRENS to be more energy-efficient in aspect of handling mobility and shortcutting. TARS utilizes a improved method, without redundant messages used in TRENS, to track mobile sinks and targets. Besides, TARS adopts a lightweight shortcutting method to decrease the number of
visited node in TRENs so that TARS decrease the length of the data packets. Moreover, TARS introduces a time-scheduling radio off method to control the radios of grid heads to reduce the energy consumption in the idle period. Finally, we compare the performance of TARS with other protocols and TRENs with extensive simulation. The simulation results show that TARS not only can successfully handle the routing issue caused by sink/target mobility but also obtains better performance than TRENs in aspect of energy consumption and data delivery.

REFERENCES


