HAR: Hierarchy-Based Anycast Routing Protocol for Wireless Sensor Networks

Niwat Thepvilojanapong
Dep. of Info. and Comm. Eng.
University of Tokyo, Japan
wat@mcl.iis.u-tokyo.ac.jp

Yoshito Tobe
Dep. of Info. Systems
and Multimedia Design
Tokyo Denki Univ., Japan
yoshito@unl.im.dendai.ac.jp

Kaoru Sezaki
Institute of Industrial Science
University of Tokyo, Japan
sezaki@iis.u-tokyo.ac.jp

Abstract

In this paper, we present Hierarchy-Based Anycast Routing (HAR), a routing protocol for collecting data over multi-hop, wireless sensor networks. The design of the protocol aims to satisfy the requirements of sensor networks that every sensor transmits sensed data to the base station periodically or spontaneously. The base station constructs hierarchical tree by finding its child nodes which in turn discover their own child nodes and so on. HAR avoids both flooding and periodic updating of routing information but the tree will be reconstructed upon node failures or adding of new nodes. By knowing only its own parent and grandparent, each sensor can make forwarding decisions regardless of the knowledge on other neighboring nodes or geographical information. We evaluate the performance of HAR by using the ns-2 simulator and comparing with those of both DSR and AODV. The simulation results demonstrate that HAR achieves much higher delivery ratio and lower latency on various scenarios.

1. Introduction

Recent advances in MEMS-based sensor technology and low-power RF and OS design have enabled the development of relatively inexpensive and low-power wireless sensors [10, 11]. A great number of such sensors can coordinate amongst themselves to achieve a larger sensing task both in urban environments and in inhospitable terrain [8]. For instance, we can use wireless sensor networks for environmental and habitat monitoring, tracking system, failure detection, and intrusion detection [17, 5, 18, 15]. However, the proclaimed limitations of sensor networks which are resource constraints including memory storage, computational power, communication bandwidth, and energy resources motivate the challenges in designing a routing protocol that fulfills the requirements of sensor networks.

Generally, a large number of sensors are deployed in remote terrain. These sensors coordinate to establish a communication network, monitor specified tasks, and report sensed data periodically or spontaneously to the nearest base station[1]. When an existing sensor is out of order due to numerous reasons, alive sensors must reorganize themselves to repair failed routes. On the other hand, the user may deploy additional sensors to mitigate a severe effect of many failed nodes, thereby enforcing the sensors to reconstruct themselves to take advantage of the added system resources. Hence, we consider a routing protocol that is based on specific communication pattern and also robust to the dynamic natures of sensor networks.

In particular, our design has been driven by the following goals.

• **Simplicity and Scalability**. Since unconstrained scale is an inherent feature of a sensor network and the sensors have limited computing capability and memory resources, we seek to minimize the number of operations performed and the states maintained at each sensor. In particular, each sensor does not maintain all neighboring nodes and the path calculation is not based on the complex algorithms such as Dijkstra's or Bellman-Ford algorithm.

• **Robustness**. The solution should provide self-organized mechanisms in order to deal with the dynamic natures of sensor networks described so far.

The contributions of the paper are as follows.

• We propose *Hierarchy-Based Anycast Routing* (HAR) protocol for data collection in multi-hop, wireless sensor networks. Each sensor in HAR can route the data to a potential nearest base station.

1 "Base station", "sink node", and "sink" are used interchangeably in the paper.

• We demonstrate that HAR works well on various scenarios through simulated networks.

The remainder of the paper is organized as follows. Section 2 describes a network model we consider. Section 3 enumerates the mechanisms of HAR in details. Section 4 evaluates the performance of HAR by the simulations. Section 5 discusses related work. Finally, we conclude by summarizing our findings and identify future research opportunities in Section 6.

2. Network Model

We consider a network composed of a small number of base stations and a numerous number of wireless sensors randomly distributed in an interesting area. These sensors have limited processing power, storage, bandwidth, and energy, while the base stations have powerful resources so as to perform any tasks or communicate with the sensors. In particular, the sensors have omni-directional antennas and use RF to communicate. We assume that the sensors are not mobile nodes, *i.e.*, all nodes are fixed for the duration of their lifetime, however, the sensor network we consider has dynamic natures as described so far.

We design a routing protocol for wireless sensor networks whose communication pattern differs from conventional mobile ad hoc networks. Let N and BS be a set of sensors and base stations respectively. HAR is a multipoint-to-point protocol for communicating parties (s, d), where $s \in \{N\}$ and $d \in \{BS\}$, namely, every sensor tries to report sensed data to the nearest base station which is a concept of *anycast* communication. Unlike HAR, previous works [21, 13, 20] are point-to-point routing protocols, *i.e.*, $s, d \in \{N, BS\}$.

3. HAR: Hierarchy-Based Anycast Routing

HAR bases on the hierarchical tree where the base stations are root nodes. Every sensor must be a member of the tree, *i.e.*, an internal or leaf node, in order to communicate with the base station and it may be act as a router if necessary. The cycle of all sensors (exclude the base stations) starts from the joining mechanism (Section 3.2) where the node attempts to attach to the tree.

The format of the packets is as follows: $\langle type$, ID_{src}, ID_{dst}, ID_{grp}, seq, len, $data\rangle$. The $type$ field is used to specify the type of the packet. The ID_{src}, ID_{dst}, and ID_{grp} fields are the source ID, the destination ID, and the group ID respectively. The group ID is an optional field, it is used to distinguish the trees created by different base stations. Thus, we can use a base station ID as a group ID. The seq field is a sequence number of the packet. The len field is the length of the packet and the $data$ field is used for carrying any data.

3.1. Building Hierarchical Tree

The base station initiates the tree construction by broadcasting a *child request* (CREQ) packets to discover the child nodes. *Nonmember*, a node which does not attach to the tree yet, determines its parent from received CREQ packets by waiting for a short period of time (T_{creq}) in order to collect a number of candidates and choose a node whose defined metric is the best one (highest received signal strength, highest remaining energy, *etc.*). The candidates are kept in a *parental candidate (PC) table* which maintains the pairs of candidate ID and metrics. A *Member* which is an internal or leaf node also updates the PC table according to an incoming CREQ packet. We can also limit the size of the PC table by deleting the stale information in the case of overflow. A nonmember chooses a parent according to the following metrics. Let us assume crq_time and $joined\_time$ denote the time that a node first received a CREQ packet and the time that a node joined the tree respectively. The nonmember chooses a node whose crq_time is the minimum and if many candidates have the same value of this metric, it will choose a node whose $joined\_time$ is the minimum. Less time implies that a candidate is nearer the base station which results in the shorter path. We decide to use time related metrics because the implementation is easy comparing to measuring the received signal strength and the amount of remaining energy. Moreover, the signal strength is affected by milieu and the remaining energy is a variable metric, *i.e.*, the energy decreases along the time.

After choosing a parent, the nonmember sends a *child reply* (CREP) packet to the selected parent so as to inform that it will be a child node. Upon receiving the CREP packet, the parent node confirms an acceptance of a new child node by replying with a *child acceptance* (CACP) packet. If the child node does not receive the CACP packet within a period of T_{cacp}, it will retransmit the CREP packet. This retransmission is performed two times. After three timeouts, it will choose a new parent from the PC table. If the PC table is empty, it will use the joining mechanism to discover a new parent (Section3.2). After receiving the CACP packet from the parent, the child node does the same process as its parent, *i.e.*, broadcasting the CREQ packet to discover its own child nodes in the next level of the tree. Note that the $joined\_time$ is the time when the node receives the CACP packet. These procedures are performed by every node throughout the network. An example of a tree constructed by HAR is shown in Fig. 4(a).

3.2. Adding the Nodes

A newly deployed node finds a parent by using a *joining mechanism* as follows. A joining node broadcasts a *parent request* (PREQ) packet making the neighboring nodes aware of its existence. Any members of the tree that hear this packet reply by unicasting a CREQ packet to the joining node. Note that this CREQ packet is same as described in Section 3.1 except that we use unicasting instead of broadcasting. Then, the processes will follow the tree construction phase, *i.e.*, the joining node sends a CREP packet to a selected parent and waits for a CACP packet as a confirmation of their relation. If the joining node does not receive any CREQ packet after broadcasting the PREQ packet, it infers that no any node is within its radio coverage or all of its neighboring nodes do not attach to the tree yet. In this case, it waits for an incoming CREQ packet after one of its neighbors has attached to the tree. As an option, joining node can broadcast the PREQ packet periodically until receiving the CREQ packets.

3.3. Dealing with Node Failures

A *Leaving mechanism* described in this section is used to reconstruct the tree if some internal nodes have failed due to numerous reasons. For instance, the battery of the node is depleted with the time, or the node can be damaged due to harsh environment or by the enemy. The tree in HAR is self-organized and it is reconstructed on-demand, *i.e.*, whenever the nodes have data to send. A detection of such failed nodes relies on the underlying MAC layer protocol. If an acknowledgement on MAC layer does not arrive, the node infers that its communicating party (which is its parent) has left from the network.

When an *orphaned node* is aware of the absence of its parent, it immediately switches to a new parent by choosing the most appropriate one from the PC table. This can be done by sending a CREP packet to a newly selected parent and waiting for a CACP packet. If there is no any candidate in the PC table, it behaves as if it is a newly deployed node by following the joining mechanism (Section3.2). However, its child and grandchild nodes will not reply to this PREQ packet to prevent routing loop. Based on all of simulations done in Section 4, two-hop information is enough for dealing with routing loop problem. Every node beneath the orphaned node does nothing because they are not aware of the absented node. They still forward packets to the orphaned node as usual, and the orphaned node keeps received packets in its buffer for sending later. In the worst case that the orphaned node does not have any candidate in the PC table and no any response to the PREQ packet, it sends a *parent query* (PQRY) packet to its child nodes asking whether they have any candidate for a parent. The child nodes reply with a *parent reply* (PREP) packet containing such information. Then, the orphaned node randomly chooses a child which has at least one candidate parent as its new parent by sending a *reverse* (REV) packet to inform a new relation, and that child will switch to a new parent chosen from the PC table. If all of its child nodes do not have any candidate parent, the orphaned node randomly chooses one child as a new parent by sending the REV packet as above and let this selected child find a new parent using the joining mechanism (Section3.2). Note that the last scenario is a rare case that may occur in highly sparse networks.

3.4. Anycast Routing and Discussions

When the network size becomes larger, it is impossible to use only one base station even though we have an optimal routing protocol because the traffic will concentrate around the base station incurring high loss rate. Thus, the user can deploy the base stations at some ratio compared to the number of sensors in order to distribute the loads. Multiple base stations can operate independently without any change in our protocol. Each node should attach to the tree created by a *potential* nearest base station because a CREQ packet from such base station should arrive first. The nodes can use the group ID to distinguish different base stations. Thereby, they can attach to multiple trees in order to achieve the robustness against failed nodes, *i.e.*, multipath routing is supported. To collect the data, each node just forwards its sensed data and all of received data to its parent. If it does not attach to the tree yet, it keeps such data in the buffer and send them later. The algorithms described thus far are summarized as pseudo-codes in Algorithm 1 and 2.

A state transition diagram is illustrated in Fig. 1. The sensor starts from a NONMEMBER state where it broadcasts a PREQ packet to discover a parent and waits for a CREQ packet at a WAIT_CREQ state. After collecting the candidates for a period of T_{creq} at a COLLECT state, it selects a parent based on the proposed metrics at a SELECT_MT state. A WAIT_CACP is a state where the node waits for a CACP packet before entering a MEMBER state. At a SELECT_RAN state, the node randomly selects one of child nodes as a new parent.

Each node in HAR relies only on the knowledge of a parent, a grandparent, and a PC table which should be small enough to keep in the node itself[2]. However, the PC table is an option because it is decomposable information. The nodes can attach to a new parent faster with

2 Mica2 has 128kB of programmable memory and 4kB of data memory.

Algorithm 1 The main algorithm of HAR protocol.

```
void main() {
$num\_crp \leftarrow 0$ // number of CREP packets sent
$flag\_prt \leftarrow$ DOWN // status of $parent$
$flag\_crq\_sent \leftarrow$ NO // status of CREQ packet sent
$flag\_crq\_rcv \leftarrow$ NO // status of CREQ packet received
$flag\_crp\_sent \leftarrow$ NO // status of CREP packet sent
$flag\_selprt \leftarrow$ NO // status of calling $select\_parent$()
BS broadcasts CREQ packet
Sensor broadcasts PREQ packet
while $rcv\_pkt$ do // $rcv\_pkt$ is a received packet
  if $rcv\_pkt$ is data packet then
    if destination is base station then
      if $flag\_prt$ = UP then
        forward $rcv\_pkt$ to $parent$
        if link-layer detects failed link then
          $link\_failed()$
        end if
      else // ($flag\_prt$ = DOWN || $flag\_prt$ = REPAIR)
        buffer $rcv\_pkt$ in a queue
      end if
    end if
  else // $rcv\_pkt$ is routing packet
    if $rcv\_pkt$ is CREQ packet then
      if ($flag\_prt$ = UP) || ($my\_addr$ = BS) then
        drop CREQ packet
      else
        $pc\_table \leftarrow (ID_{src}, crq\_time, joined\_time)$
        if $flag\_crq\_rcv$ = NO then
          $crq\_time \leftarrow$ CURRENT_TIME
          $flag\_crq\_rcv \leftarrow$ YES
        end if
        if $flag\_selprt\_call$ = NO then
          $flag\_selprt\_call \leftarrow$ YES
          call $select\_parent()$ at $T_{crq}$ seconds later
        end if
      end if
    else if $rcv\_pkt$ is CREP packet then
      if ($flag\_prt$ = UP) || ($my\_addr$ = BS) then
        send CACP packet
      end if
    else if $rcv\_pkt$ is CACP packet then
      $flag\_crp\_sent \leftarrow$ NO
      $num\_crp \leftarrow 0$
      $joined\_time \leftarrow$ CURRENT_TIME
      send all buffered packets in queue to $parent$
      if ($flag\_prt$ = DOWN) & ($flag\_crq\_sent$ = NO) then
        broadcast CREQ packet
        $flag\_crq\_sent \leftarrow$ YES
      end if
      $flag\_prt \leftarrow$ UP
    else if $rcv\_pkt$ is PREQ packet then
      if ($my\_addr$ = BS) || (($flag\_prt$ = UP) & ($ID_{src} \neq parent$)
      & ($ID_{src} \neq$ grandparent node)) then
        unicast CREQ packet
      end if
    else if $rcv\_pkt$ is PQRY packet then
      send PREP packet
    else if $rcv\_pkt$ is PREP packet then
      randomly choose a new parent from its children
      send REV packet to selected parent
    else if $rcv\_pkt$ is REV packet then
      if $pc\_table$ is not empty then
        $select\_parent()$
      else
        broadcast PREQ packet
      end if
    end if
  end if
end while
}
```

the help of the PC table but the information in the PC table may be stale, *i.e.*, a newly selected parent is also a left node. If the nodes always broadcast the PREQ packet to discover a new parent without relying on the PC table, it

Algorithm 2 The functions called by the main algorithm.

```
void sendChdRep() {
send CREP packet to $parent$
$flag\_crp\_sent \leftarrow$ YES
$num\_crp \leftarrow num\_crp + 1$
call $wait\_cacp()$ at $T_{cacp}$ seconds later
}
void select_parent() {
if BS is in $pc\_table$ then
  $parent \leftarrow$ BS
else
  for all members in $pc\_table$ do
    $parent \leftarrow$ node whose $crq\_time$ is the minimum
    if $crq\_time$ is equal then
      $parent \leftarrow$ node whose $joined\_time$ is the minimum
    end if
  end for
end if
$sendChdRep()$
}
void wait_cacp() {
if $flag\_crp\_sent$ = YES then
  if $num\_crp > 3$ then
    $pc\_table \leftarrow pc\_table - parent$
    $parent \leftarrow$ NULL
    $num\_crp \leftarrow 0$
    if —$pc\_table| > 0$ then
      $select\_parent()$
    else
      periodically broadcast PREQ packet until getting CREQ packet
    end if
  else // $num\_crp \leq 3$
    $sendChdRep()$
  end if
end if
}
void link_failed() {
buffer packets in queue
$flag\_prt \leftarrow$ REPAIR
$pc\_table \leftarrow pc\_table - parent$
$parent \leftarrow$ NULL
if $pc\_table > 0$ then
  if $crq\_time$ of at least one member in $pc\_table$ is less than own $crq\_time$
  then
    $select\_parent()$
  else
    broadcast PREQ packet
    $num\_crp \leftarrow 0$
  end if
else // $pc\_table$ = NULL
  broadcast PREQ packet
  $num\_crp \leftarrow 0$
end if
}
```

will get fresh information. Since the state required on each node is constant (a fix-sized PC table) independent of node density and network size, HAR is highly scalable. Every node in HAR broadcasts only once to discover the route, thereby no propagation of routing packets issued by each node throughout the network. In other words, routing packets are limited to one-hop neighboring nodes. Moreover, HAR does not apply periodic updating which reduces traffic load so much. Furthermore, geographical information is not necessary.

4. Performance Evaluation

To evaluate the performance of HAR, we use the *ns*-2 [1] simulation tool to run a number of simulations de-

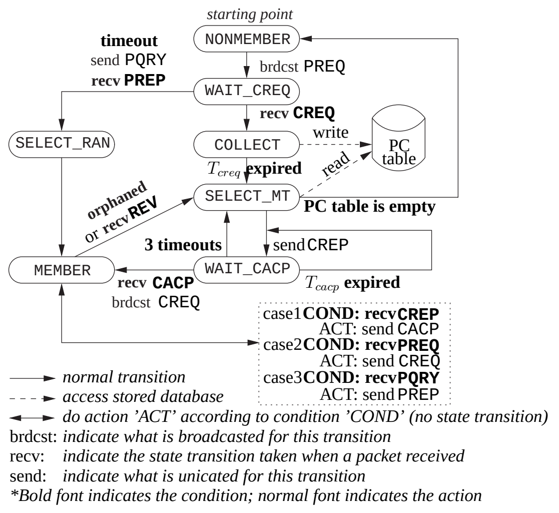


Figure 1. State transition diagram.

scribed in this section. We compare the performance with two well-known ad hoc routing protocols, Dynamic Source Routing (DSR) [13] and Ad-Hoc On-Demand Distance Vector (AODV) [20] protocol, which have been shown to offer promising performance [4, 6].

4.1. Methodology

The ns-2 simulator includes full simulation of the IEEE 802.11 physical and MAC layers. Our simulations use this MAC layer and assume symmetric links. Using this MAC layer does not affect the evaluation because we need to evaluate the network layer of three protocols. We randomly placed 50, 70, and 100 sensors in a 250m by 250m square regions. Each node has fixed radio coverage of 50 meters. The nodes have fixed positions without any movement for the entire simulation, *i.e., immobile* nodes. We use constant bit rate (CBR) as our traffic sources. These CBR sources send 64-byte data packets at the rate of 0.25, 0.5, 1, and 2 packets per second, *i.e.*, 128, 256, 512, and 1,024 bps respectively, while the bandwidth of sensors is set to 19.2 kbps. The CBR agent will be attached to a UDP agent, which in turn attached to the source node. To study the impact of dynamics, we deploy new nodes and make battery of the existing nodes deplete around the half-way of the simulations.

For all simulations, the communication patterns are peer-to-peer and the starting time of each connection is randomly selected. One node from each simulation is randomly chosen as a base station and it is the only destination for all traffic sources, while all remaining nodes (*i.e.*, 49, 69, and 99 nodes) are source nodes (one flow per one source). We let a routing agent start to construct the tree at the beginning of the simulation. Each simulation is run for 300 simulated seconds. Each data point represents an average of ten runs with identical traffic models, but different randomly generated topologies. This excludes 100-node networks where each data point is an average of five runs.

Since the performance of AODV is worse than DSR as being shown in the following results, we ran only 50-node network for AODV to save simulation resource and time. We also ran same simulations on DSDV routing protocol [21] which shows worse performance than AODV and DSR. Therefore, we do not include the results of DSDV in this paper. The parameters of HAR used in the simulations are set as follows: T_{creq} = 0.1 second, and T_{cacp} = 0.3 second.

4.2. Performance Metrics

To compare with other protocols, we choose to evaluate them according to the following metrics.

- *Packet delivery ratio* (PDR): the ratio between the number of data packets received by the destination and the number of data packets sent by the source.
- *Average Latency*: the average end-to-end delay observed between transmitting a data packet and receiving it at the destination.
- *Average path length*: the average number of hops a data packet took to reach its destination.

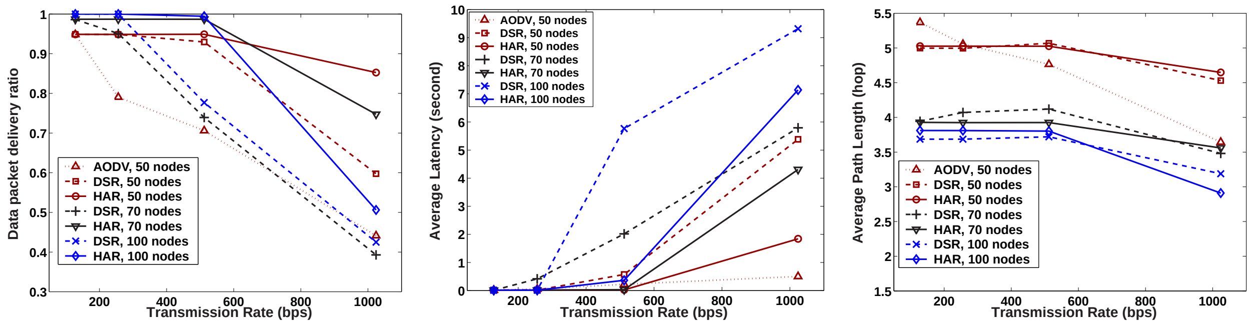
Packet delivery ratio is important as it shows the loss rate seen by the transport protocols. It also affects the maximum throughput that the network can support. This throughput can be investigated by increasing the transmission rate. This metric characterizes both the completeness and correctness of the routing protocol.

Average latency is an important metric for comparing as it measures the quality of path decided by routing algorithm. The protocols that send large numbers of routing packets can also increase the probability of packet collisions and may delay data packets by queuing them in the buffer.

Average path length measures the ability of the routing protocol to efficiently use network resources by selecting the shortest path from a source to a destination.

4.3. Performance Comparison in Static Networks

The first set of the simulations uses differing node densities with four offered loads. In particular, we randomly deploy 50, 70, and 100 nodes in the same size of square region. The comparisons of three performance metrics are illustrated in Fig. 2. In Fig. 2(a), the PDR of AODV dramati-



(a) Data packet delivery ratio. (b) Average latency. (c) Average path length.

Figure 2. Performance comparison in static networks.

cally drops as the offered load is increased. This is a reason that we exclude AODV from the simulations with higher node density. The PDRs of DSR and HAR are comparable at lower offered load where both can deliver more than 95% of the originated data packets. However, HAR clearly outperforms DSR at the higher offered load. With 70-node network, for example, HAR delivers 25% and 35% more than DSR at 512- and 1,024-bps offered load respectively. At high offered load, the data packets drop due to high congestion at the nodes around the sink. When a node cannot deliver a packet to the next hop due to congestion, it implies the absence of the next-hop node. In such case, DSR and AODV will flood route request (RREQ) packets to discover a new route. These control packets lead to a chaos of the network which in turn incurs more congestion. In contrast, HAR discovers a new route by broadcasting which is limited to only one hop, thereby, HAR is more tolerable to offered load than DSR and AODV.

When we consider the impact of node density in the same figure, the PDRs of 70- and 100-node network are higher than 50-node network at the light load because the main cause of dropped packets is partitioned network. Intuitively, sparse network has higher probability of partitioned network than dense network. As one would expect, dense network has lower PDR than sparse network due to high congestion around the sink as described above. With DSR, surprisingly, the PDR of 100-node network is higher than 70-node network. When we investigate in the details, DSR can deliver only 65% of the originated packets in the 6^{th} topology which does not include in 100-node network. Note again that the results are an average of ten topologies except 100-node network which is an average of five topologies.

Average latencies of the above scenarios are shown in Fig. 2(b). Three of the protocols for all node densities have comparable latency (less than 60 ms) in lower traffic load except DSR in 70-node network which takes up to 400 ms. In particular, HAR takes the shortest period of time amongst the three protocols in delivering data to the sink. At the higher offered load, HAR demonstrates significantly lower latency than DSR for all of node densities. AODV seems to show outstanding performance at the highest offered load, however, over 50% of data packets have lost (Fig. 2(a)), especially the packets destined to the distant destination. Very high latency of DSR is a result of its attempt in trying to deliver the data packets on a distant route through high congestion occurring around the sink.

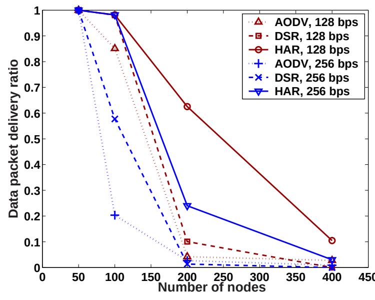


Figure 3. PDR when varying network size.

In all cases, HAR and DSR have comparable path length (Fig. 2(c)). AODV traverses longer path at lower offered load but it takes shorter path at the higher load due to high dropped packets destined to distant route as described so far. An example of a tree constructed by HAR on the first topology of 70-node network is illustrated in Fig. 4(a).

4.4. Impact of Network Size

Next, we study the effect of network size on the performance of each protocol. We evaluate only PDR when deploying 50, 100, 200, and 400 nodes in 250m by 250m,

		PDR		Latency (s)	
	Load (bps)	512	1024	512	1024
JOIN:	HAR	0.95	0.78	0.33	2.45
	DSR	0.90	0.64	0.97	5.19
	AODV	0.70	0.39	0.27	0.60
LEAVE:	HAR	0.92	0.78	0.08	1.16
	DSR	0.90	0.66	0.94	3.71
	AODV	0.75	0.45	0.21	0.48

Table 1. Performance comparison in dynamic networks.

350m by 350m, 500m by 500m, and 700m by 700m square regions respectively. In other words, we try to keep node density constant for different network sizes. We simulate only 128-bps and 256-bps scenarios on one random topology due to the lack of processing resource. The results on both offered loads shown in Fig. 3 are identical, *i.e.*, HAR is more scalable than both DSR and AODV. In particular, HAR still delivers data at an acceptable ratio (more than 98%) on 100-node network with 256-bps offered load. The user may deploy additional sinks in order to distribute network loads and HAR can automatically forward sensed data to the nearest sink as being shown in Section4.6.

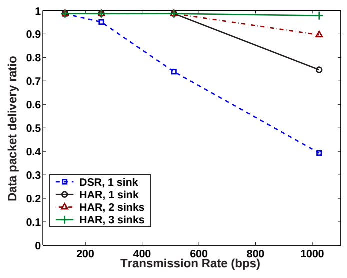
4.5. Impact of Network Dynamics

We also simulate dynamic scenarios of sensor networks, *i.e.*, joining and leaving scenarios. To simulate the joining scenarios, 10 nodes are randomly deployed at the 150^{th} second in addition to 50 nodes deployed from the beginning of the simulations. For the leaving scenarios, we deploy 60 nodes at the beginning of the simulations and apply an energy model by providing much enough energy for 50 nodes and making the battery of 10 nodes depletes at some point of time before the simulations end. In particular, the battery of such leaving nodes depletes around 100^{th}–120^{th} second depending on the offered load. We choose heavy offered loads (*i.e.*, 512- and 1,024-bps load) to make the situation fairly challenging for the routing protocols. These experiments are used to evaluate the robustness and resilience of the protocol. The results are presented in Table 1.

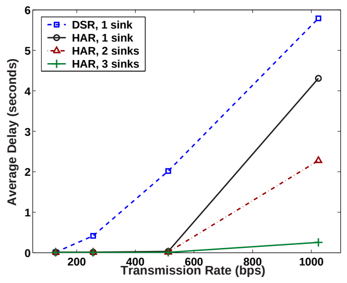
The results are identical with the static scenarios that HAR has better PDR and latency than DSR which in turn performs better than AODV. The exception is low delay of AODV due to high losses as explained in the static networks.

4.6. Impact of the Number of Sinks

The last set of the experiments uses differing number of sinks ranging from one to three nodes. Since 50-node network may be too sparse, *i.e.*, 17 nodes per sink in average



(a) Data packet delivery ratio.



(b) Average latency.

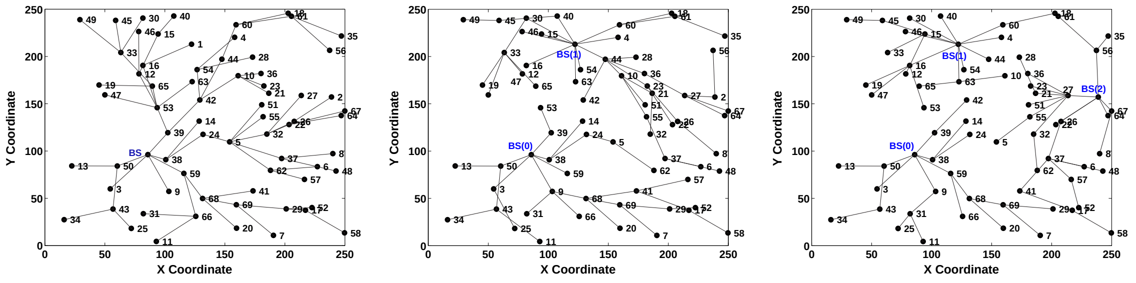
Figure 5. Performance comparison on 70-node network when varying the number of sinks.

if there are three sinks, we choose 70-node network for the experiments. Thus, DSR is only one comparative protocol and we run only 1-sink scenarios for DSR because the sensors do not know the address of the nearest sink in advance and DSR must specify destination address when flooding the RREQ packets. We also do not provide the address of the nearest sink for HAR, but it can discover a *potential* nearest sink by itself. The PDR and latency are shown in Fig. 5 and the examples of the trees created by HAR are illustrated in Fig. 4.

As expected, we can achieve higher PDR and lower latency because congestion as well as the path length are reduced. HAR can deliver at least 98% of the originated data packets (23% increasing from 1-sink scenario) when the highest load are offered. Although we have an optimal routing protocol, increasing the number of sinks is necessary for large-scaled sensor networks.

5. Related Work

A number of routing protocols [2] has been developed to provide efficient and robust communication in mobile ad hoc networks (MANETs) which is different from wireless sensor networks (WSNs) in an issue that the nodes are mobile. Moreover, the scale of MANETs is typically



(a) 1 sink node: BS. (b) 2 sink nodes: BS(0) and BS(1). (c) 3 sink nodes: BS(0), BS(1), and BS(2).

Figure 4. Hierarchical tree created by HAR. X- and y-axis represent node location.

much smaller compared to WSNs, and these proposals typically assume a much smaller network size. However, many MANET routing protocols can work well in immobile networks [4]. To provide the robustness against changing topology, the proactive routing protocols must periodically update states to follow current physical topology [21]. On the other hand, the reactive approaches try to reduce the cost of periodic updating by discovering the route on-demand [13, 20]. However, flooding used in these protocols still incurs congestion shown in our simulation results and it takes time to discover an available route. Therefore, we propose an alternative which is more light-weight to perform specific communication required in WSNs. Every node in HAR broadcasts only once to discover the route which obviously incurs less overhead than flooding.

Location-based routing can apply well to WSNs. LAR [16] and GPSR [14] are geographic routings that use location information to decrease the overhead of route discovery and find the routes quickly. Unfortunately, such approaches may not implement into many sensor networks because each sensor requires to know its exact geographic location. Current methods of determining geographic location [3, 7, 22] consume much energy and may not be possible in many sensor network scenarios. Moreover, location-aware device increases the production cost for sensors, especially in large-scaled sensor networks.

Directed diffusion [12] is a data-centric routing protocol based on the name of data. The sinks draw interesting information by flooding the interests and setting up the gradients within the network. It is query-style protocol dealing with the name of data which is different from our work that aims to support periodic monitoring applications using WSNs.

Randomized algorithm is used in constrained random walk [23] to determine the next hop in order to achieve load balancing. However, they consider only one-source network in their evaluation because multiple-source network requires more complex computation to balance energy. In contrast, HAR considers multiple sources and sinks.

VPCR (Virtual Polar Coordinate Routing) [19] is a point-to-point routing protocol based on hierarchical tree as our work. To create the tree, each node must know its location. Moreover, each node must keep two-hop neighbors in order to achieve the best performance. Adding and removing nodes tend to adversely affect the tree because VPCR need to preserve the alignment with physical topology. In the worst case, the subtree must be rebooted. The tree in HAR does not depend on the location of nodes that makes protocol more robust to dynamic networks.

Ye *et al.* proposed Minimum Cost Forwarding algorithm for collecting data in sensor network [25]. Each node maintains the least cost estimate from itself to the base station. Data packets are forwarded by broadcasting which does not guarantee reachability. Moreover, it wastes energy because broadcast packet is received by every neighboring node. To deal with node failures, the authors proposed to increase the cost budget at the source node which results in non-optimal path. Another simple solution is to refresh the cost field which consumes both energy and time.

LEACH (Low Energy Adaptive Clustering Hierarchy) [9] is a multipoint-to-point communication protocol aiming to achieve energy efficiency. Cluster head is self-elected and randomized rotation of cluster heads is employed to balance energy consumption. However, the authors do not consider the dynamics of network. Moreover, the selection of cluster head is optimized by some probability which does not respect to geographic location. As a result, there is a possibility that cluster heads could be concentrated in one part of the network.

Woo et al. introduced a many-to-one routing based on link connectivity statistics [24]. This kind of information must be estimated on-line requiring the node listens for the packets that are not necessarily addressed to it. This estimation comes at computational and communication cost as well as memory storage.

6. Conclusions

This paper has demonstrated that HAR efficiently collects the data packets across multi-hop wireless sensor networks while maintaining a constant amount of local state and making only local decisions. HAR also provides the solutions against dynamic natures of sensor networks. In other words, it is self-organized protocol according to the joining and leaving nodes. One great advantage of HAR which differs from conventional MANET routing protocol is that multiple base stations can independently work together and each sensor can automatically discover a potential nearest base station. We have examined the performance of HAR in terms of packet delivery ratio, latency, and path length. The results have shown that HAR achieves notably high delivery ratio, low latency, and comparable path length with the existing protocols. Besides, it is more tolerable to high offered loads than the existing solutions. One application in sensor networks that incurs high traffic load is structure health monitoring (SHM) [18, 15]. Nonetheless, the actual traffic rate is greater than the data generation rate of a node due to forwarded traffics, especially around the sink.

However, we have not explored all required performance metrics such as energy consumption which is one of our future works. Based on our preliminary results, AODV consumes higher energy than HAR. We plan to employ a simple aggregation algorithm in HAR to see the improvement of the performance. Another research direction is to include the reliability in our protocol. We also plan to implement HAR in Mica Mote to study its performance in the real world environment.

References

[1] Network simulator - ns-2. http://www.isi.edu/nsnam/ns/.

[2] M. Abolhasan, T. Wysocki, and E. Dutkiewicz. A review of routing protocols for mobile ad hoc networks. *Ad Hoc Networks*, 2(1):1–22, Jan. 2004.

[3] P. Bahl and V. N. Padmanabhan. RADAR: an in-building rf-based user location and tracking system. *Proc. of IEEE INFOCOM*, pages 775–784, Tel-Aviv, Israel, Mar. 2000.

[4] J. Broch et al. A performance comparison of multi-hop wireless ad hoc network routing protocols. *Proc. of MOBICOM*, pages 85–97, Dallas, Texas, Oct. 1998.

[5] A. Cerpa et al. Habitat monitoring: application driver for wireless communications technology. *SIGCOMM Comput. Commun. Rev.*, 31(2 supplement):20–41, 2001.

[6] S. R. Das, C. E. Perkins, and E. M. Belding-Royer. Performance comparison of two on-demand routing protocols for ad hoc networks. *IEEE INFOCOM*, pp. 3–12, Mar. 2000.

[7] L. Doherty, K. S. J. Pister, and L. E. Ghaoui. Convex optimization methods for sensor node position estimation. *Proc. of IEEE INFOCOM*, pages 1655–1663, Apr. 2001.

[8] D. Estrin, R. Govindan, J. S. Heidemann, and S. Kumar. Next century challenges: Scalable coordination in sensor networks. *Proc. of MOBICOM*, pages 263–270, Aug. 1999.

[9] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. *Proc. of Hawaaian International Conference on Systems Sciences*, pages 3005–3014, Jan. 2000.

[10] J. Hill and D. Culler. Mica: a wireless platform for deeply embedded networks. *IEEE Micro*, 22(6):12–24, 2002.

[11] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister. System architecture directions for networked sensors. *Proc. of ASPLOS*, pages 93–104, Nov. 2000.

[12] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed Diffusion: a scalable and robust communication paradigm for sensor networks. *Proc. of MOBICOM*, pp. 56–67, 2000.

[13] D. B. Johnson, D. A. Maltz, and J. Broch. DSR: The dynamic source routing protocol for multi-hop wireless ad hoc networks. In C. E. Perkins, editor, *Ad Hoc Networking*, chapter 5, pages 139–172. Addison-Wesley, 2001.

[14] B. Karp and H. T. Kung. GPSR: greedy perimeter stateless routing for wireless networks. *Proc. of MOBICOM*, pages 243–254, Boston, Massachusetts, Aug. 2000.

[15] S. Kim, D. Culler, and J. Demmel. Structural health monitoring using wireless sensor networks, 2003. http://www.eecs.berkeley.edu/∼binetude/course/cs294_1/paper.pdf.

[16] Y.-B. Ko and N. H. Vaidya. Location-aided routing (LAR) in mobile ad hoc networks. *MOBICOM*, pp. 66–75, 1998.

[17] A. Mainwaring et al. Wireless sensor networks for habitat monitoring. *Proc. of WSNA*, pages 88–97, Sept. 2002.

[18] K. Mechitov, W. Kim, G. Agha, and T. Nagayama. High-frequency distributed sensing for structure monitoring. *Proc. of INSS*, Tokyo, Japan, June 2004.

[19] J. Newsome and D. Song. GEM: Graph embedding for routing and data-centric storage in sensor networks without geographic information. *Proc. of SenSys*, pp. 76–88, 2003.

[20] C. E. Perkins, E. M. Belding-Royer, and S. Das. Ad hoc on-demand distance vector (AODV) routing. RFC 3561, IETF, July 2003.

[21] C. E. Perkins and P. Bhagwat. Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. *Proc. of ACM SIGCOMM*, pp. 234–244, Sept. 1994.

[22] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan. The cricket location-support system. *Proc. of MOBICOM*, pages 32–43, Boston, Massachusetts, Aug. 2000.

[23] S. D. Servetto and G. Barrenechea. Constrained random walks on random graphs: Routing algorithms for large scale wireless sensor networks. *Proc. of WSNA*, pages 12–21, Atlanta, Georgia, Sept. 2002.

[24] A. Woo, T. Tong, and D. Culler. Taming the underlying challenges of reliable multihop routing in sensor networks. *Proc. of SenSys*, pages 14–27, Los Angeles, CA, Nov. 2003.

[25] F. Ye, A. Chen, S. Lu, and L. Zhang. A scalable solution to minimum cost forwarding in large sensor networks. *Proc. of ICCCN*, pages 304–309, Scottsdale, AZ, Oct. 2001.