Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2013, Article ID 958061, 14 pages
http://dx.doi.org/10.1155/2013/958061

Research Article

Architecture and Routing Protocols for Smart Wireless Home Sensor Networks

Yang Xu,[1] Shuai Wu,[1] Ruochen Tan,[1] Zheng Chen,[1] Min Zha,[2] and Tina Tsou[2]

[1] *University of Electronic Science and Technology of China, Chengdu, Sichuan 611731, China*
[2] *Huawei Technologies Co., Ltd., Shenzhen, Guangdong 518129, China*

Correspondence should be addressed to Yang Xu; xuyang.uestc@gmail.com

Received 25 May 2012; Revised 29 September 2012; Accepted 18 October 2012

Academic Editor: Regina B. Araujo

As an important application domain of wireless sensor networks (WSN), wireless home sensor network (WHSN) can be built as a traditional WSN. However, when we consider its own character that plug-in sensors are fixed with AC power supply while mobile sensors are battery powered, traditional WSN techniques do not match well. In this paper, we propose a smart wireless home sensor network architecture with improved routing protocols. It is a hierarchical architecture in which AC-powered sensors act as backbone nodes for data retransmission, while battery-powered sensors act as leaves that only transmit data relevant to themselves. Each sensor is assigned with a prime number as its location identifier. For our routing algorithm, the LID is used as routing address and is decomposed to a sequence of prime numbers that indicates the route towards its destination. In addition, we improve existing routing algorithms such as SPIN, LEACH, and DD to incorporate traditional WSN routing algorithms into our smart WHSN architecture, and comparable efficiencies are made between them. Moreover, we propose a network path recovery algorithm for failures that are caused by node mobility or backbone node failures. All our algorithms have been provided with faithful simulations to verify the feasibility and efficiency.

1. Introduction

Wireless home sensor networks (WHSNs) that connect household appliances to the Internet become important application domains of wireless sensor network (WSN) [1, 2]. The objective is to use kinds of wireless techniques to interconnect household electrical devices and the sensors to form a unified platform so that all smart household devices can communicate and be controlled cooperatively, securely, and robustly. There have been extensive studies made in this field to build an autoadaptive and green wireless home sensor network. Azim and Islam [3] proposed a hybrid LEACH [3, 4] algorithm to build a relay node in its hierarchical clusters so that an energy-saving WHSN can be realized. Park and Corson [5] proposed a highly adaptive distributed routing algorithm for WHSN by using the SPIN [5] protocol. Zhang et al. [6] proposed an energy efficient approach called directed diffusion (DD) [6] protocol based on passive clustering. Although these solutions have been proven to be more efficient than traditional WSN protocols when applied in home sensor networks, they only took partial advantages of the WHSN features and made unnecessary assumptions such as full mobility of WHSN nodes. Therefore, based on the state of the art analysis, a comprehensive approach that matches the needs of smart wireless home sensor network is still on the way.

In this paper, we investigated the key features of WHSN that distinguish it from traditional WSNs and proposed a novel algorithm to make a full use of those features. First of all, some nodes of WHSN are embedded in household appliances and connect AC as power supply, which are not similar with typical battery-powered WSN nodes. Hence, energy consumption is no longer a constraint for AC-powered nodes, and they might be able to make more contribution on packet transmissions than those energy-constrained mobile nodes. Secondly, the deployment of WHSN is partially fixed other than traditional WSNs where most nodes are randomly scattered. For example, temperature sensors in WHSN are adhered on the wall or embedded into the household appliances that are permanently immovable. Therefore, we can

model the WHSN architecture as a semistructured layout rather than a typical mobile ad hoc network. Thirdly, sensor nodes in WHSNs [7, 8] typically range from 100 to 1000 in a household, which is categorized as a middle size WSN. Therefore, scalability is not a major bottleneck in network design for WHSNs.

By making a good use of these key features to build a flexible and efficient wireless home sensor network, we proposed a hierarchical architecture for WHSN. In this architecture, sensor nodes are categorized into two classes: plug-in nodes that are energy restriction free but localization fixed and energy-restricted nodes that are modeled with mobilities. According to our design, nodes are organized as a tree structure where the gateway acts as the root. To take the advantage of plug-in nodes, we deployed them as backbones of the tree for data retransmissions, and their immovability is helpful to keep the tree structure stable. As energy-restricted nodes are normally movable, such as sensors in a robot, they are more suitable to be connected as leaf nodes in the network.

Based on the architecture design, we are able to customize simple and flexible routing protocols for WHSN. The key innovation is that we excluded the traditional routing table, by assigning each node with a prime-numbered address, and each routing path is identified with a location identifier (LID) which is the production of the prime addresses of all the relay nodes. The key advantage of this design is that when a node chooses the next relay node, it can easily adopt a decomposing operation that only the next transmitter can be divided. When the package is transmitted, the LID is replaced with the quotient to identify the remaining path, and when the LID is a prime number, it denotes that it is the address of the destination node. Therefore, the path can be easily discovered by the relay nodes without routing table.

In addition to our new algorithm design, we incorporated traditional WSN protocols into our smart home sensor networks architecture by improving some popular routing algorithms such as SPIN, LEACH, and DD [6, 9]. The key is to take the advantage of both the hierarchical architecture and energy restriction free plug-in nodes. In the end, we introduced our network maintenance algorithm. Benefited by the WHSN features, our protocols can improve the routing efficiency and effectively reduce the cost for network reorganization caused by the movement of mobile nodes or sensor node failures.

2. Smart WHSN Architecture

In this section, we introduce our semistructured WHSN architecture. Our design is based on the key features on where and how sensors are deployed in a household. The application domain can be specifically illustrated as follows:

(1) there is a home gateway acting as the sink node to the internet;

(2) some nodes are embedded in household appliances and directly connected with AC adapter. Due to its cable connection, it is hardly movable;

(3) the other nodes are most likely moveable, such as sensors in a robot. Due to their mobilities, they are battery powered, and energy consumption is a bottleneck for data transmission;

(4) wireless home sensor networks are typically with hundreds of sensors.

2.1. Plug-In Nodes and Mobile Nodes. In a household, plug-in nodes are normally location fixed and integrated into household appliances with AC supply. For example, in Figure 1, the refrigerator contains temperature sensors and radio frequency (RF) sensors and they are all AC powered, which means they have no energy consumption limitations and can keep awake all the time. In WHSN, energy consumption is key factor to affect the lifetime of the network. If we can make use of plug-in nodes as backbone for data transmission, we can significantly reduce energy consumption of mobile nodes and extend the WHSN lifetime. In addition, plug-in nodes with power supply are normally more stable than mobile nodes. If they are used as backbone nodes, it will help improve the stability of WHSN from power-off failures.

Other than plug-in nodes, there are plenty of sensors that are movable and powered with battery. Typical mobile sensors include video cameras in robots and localization sensors in vacuum cleaners. The mobility feature of mobile sensors will make necessary changes on network topology, and as they are always battery powered, energy constraint is a key bottleneck for them involving data retransmissions. For example, as shown in Figure 2, modules of the mobile node such as the data acquisition, control, movement, and data transmission are all powered with battery.

2.2. WHSN Network Design. The key of our WHSN architecture design is to make the advantage of features that discriminates WHSNs from traditional WSNs. As explained previously, plug-in nodes are more suitable as backbone nodes to take more responsibility of data transmission according to their robustness and battery free features. Moreover, the immovability feature of the plug-in nodes help to keep the backbone stable. On the other hand, mobile nodes powered by battery should connect with the network loosely so that the network will not be significantly changed when they swift their positions, and most importantly, less data retransmissions via mobile nodes will extend their battery lifetime.

According to WHSN features, we proposed a hierarchical architecture for wireless home sensor networks as shown in Figure 3. In our design, nodes are organized as a tree, and the gateway is set as the root of the tree. In this tree, upper layers are most likely to be composed of fixed plug-in sensor nodes, while battery-powered nodes or mobile nodes are more likely to be set as the leaf nodes. Staring with an existing root, the network construction process can be briefly described in Algorithm 1. In this algorithm, nodes are recursively joined to the tree. Firstly, the pending node evaluates its own capability according to its energy constraint, bandwidth, and so forth (line 2). Next, it enumerates its neighbors, queries the capabilities of the neighbors, and reorders its neighbors by their capabilities (lines 3 and 4). With the ordered neighbor list, a node which is of the least capability than the pending

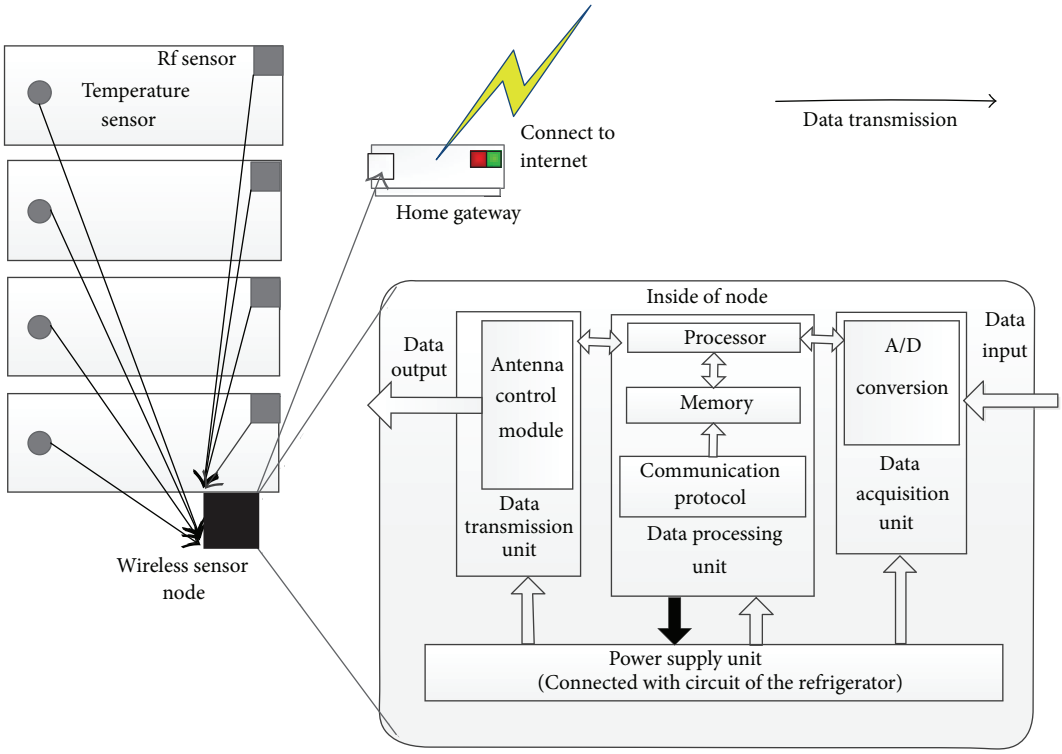


Figure 1: Smart refrigerator, an example of AC-powered plug-in nodes.

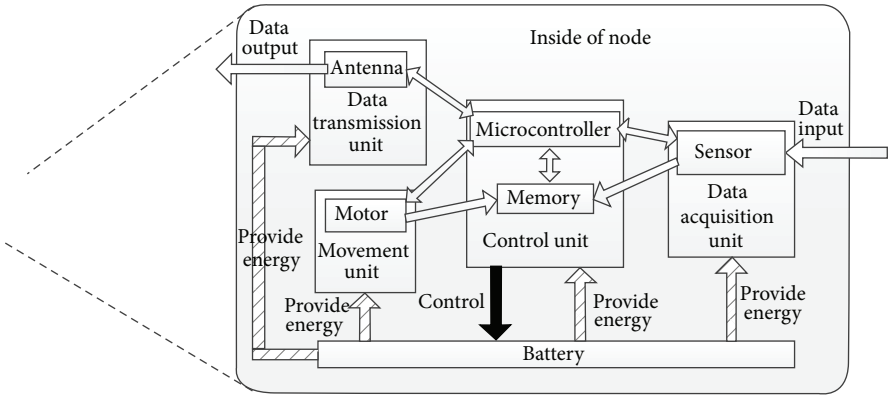


Figure 2: Smart cleaner, an example of battery-powered mobile nodes.

one and does not reach its maximum children number can be found (lines 5–9). As the last step, the pending node will be allocated with a prime-numbered address (line 10) and its LID (line 11). The addressing process will be described in the next section.

Following our network architecture design, we will describe our system design. Although WHSN is a typical WSN where popular WSN protocols can be applied straightforward, we can make necessary revisions to enhance its efficiency. As shown in Figure 4, there are four layers. Based on our hierarchical architecture in the lowest layer, we proposed a prime-numbered addressing protocol and setup for all sensors to form a logical WHSN. In routing protocol layer, we can either modify the traditional routing protocols such as SPIN, LEACH, and DD to match the feature of WHSN or build our new routing protocol as D-HiPr. As the top layer, network maintenance is key protocol to keep the mobile feature of WHSN.

3. Addressing and Routing Algorithms

When sensors in WHSN can be organized as a hierarchical architecture as we have proposed, the key is how we can design the addressing and routing protocols that take the merits of tree-like organization as the features of plug-in and mobile nodes. In this section, we present our novel algorithm of dynamical hierarchical routing protocol with prime numbers (D-HiPr) addressing.

```
(1)  for each pending node_i do
(2)      Cap ← node_i · getCap();
(3)      neighbors ← node_i · getNeighbors();
(4)      neighbors ← sortbyCap(neighbors);
(5)      parent ← getCloseCapNb(neighbors, Cap);
(6)      while getChildNum(parent) ⩾
         MaxAllowed do
(7)          neighbors · delete(parent);
(8)          parent ← getCloseCapNb(neighbors);
(9)      end while
(10)     node_i · parent ← parent;
(11)     node_i · address ← assignAddress();
(12)     node_i · LID ← assignLID(parent · LID);
(13) end for
```

Algorithm 1: Network construction.

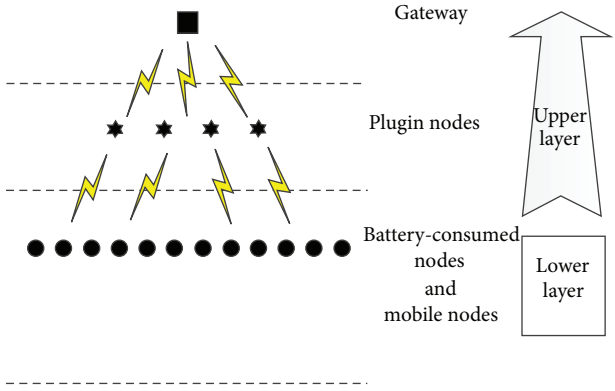


Figure 3: Hierarchical network design for WHSN.

3.1. Prime-Numbered Addressing Protocol. In our approach, each node in WHSN is assigned with a prime number as its short address, and any path can be identified by the location identifier (LID), which is the production of the prime addresses of all nodes in the path. The advantage of D-HiPr with prime number addressing lies on the following definition.

Lemma 1. *The product of all prime numbers in set P can only be divided by the product of prime numbers which are the subset of P.*

Therefore, each LID can be decomposed to a sequence of prime numbers that identifies a unique path in the WHSN tree from the root, if all the intermediate nodes in this path have different prime-numbered addresses. When a node checks who is the next relayer, it can simply adopt a decomposing operation with its children that only the address of the next transmitter could be divided. In our D-HiPr protocol, two parameters are required for a given node: a prime numbered 16-bit address A_p and LID Q which is the product of nodes in the path linking it from the root. For example, if the path from a node c to the root is $A_1, A_2, \ldots, A_c$, the LID of this node is $Q_c = A_1 \cdot A_2 \cdot \cdots \cdot A_c$. The location of a node can be identified by its parent node, and the node can get its path from its own LID. If the LID of its parent node is Q_p, the LID of node c is calculated as $Q_c = A_c \cdot Q_p$. For example, as shown in Figure 5, LID $1955 = 1 \times 5 \times 17 \times 23$ explicitly identifies the path from the root to node 23. When node 5 gains the LID, it will only find node 17 as its next relayer since only 17 can divide the LID.

In order to avoid address conflicts, the gateway responses of all the address assignment. The allocation process is briefly shown as Figure 6. There are three major phases. The first is the neighbor discovery (ND) process [10], which includes router solicitation (RS) and router advertisement (RA). The second phase is for node registrations including two new ICMPv6 messages: node registration (NR) and node confirmation (NC) [11]. Unlike the traditional multi-hop registration, NC includes a location option to contains the LID of its parent node so that its children can calculate their own LID by producing its prime-numbered address with their parent's LID. The third phase is for location registration. The node will use location registration (LR) and location confirmation (LC) messages to inform its LID to the parent node as well as the gateway. Each node in our design uses "hello" message to check whether the neighbors keep the positions. When a node is moved, it will ask the gateway to rejoin the network with a new address. If a node detects that its parent node is gone or a child is moved, network maintenance will start and we will describe that in Section 5.

3.2. Routing Protocol of D-HiPr. In this section, we represent the details of D-HiPr and how the prime-numbered address works. For example, when a packet is required to be sent from node C to the destination node D, the D-HiPr routing protocol is explicitly described as Algorithm 2. In this case, the parent node of C is P and its children are k_i ($i = 1, 2, 3, \ldots, n$) where n is the number of its children. The LIDs of node C and node D are Q_c and Q_d, and those of node P and their children node k_i are Q_p and Q_{k_i}, respectively. To route the packet in node C, the difference between the LID of the destination and its source node is obtained first

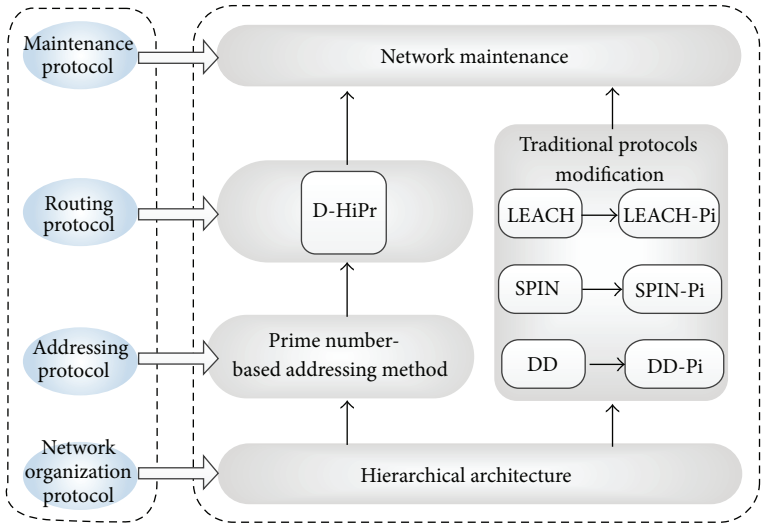


Figure 4: System architecture of a smart WHSN.

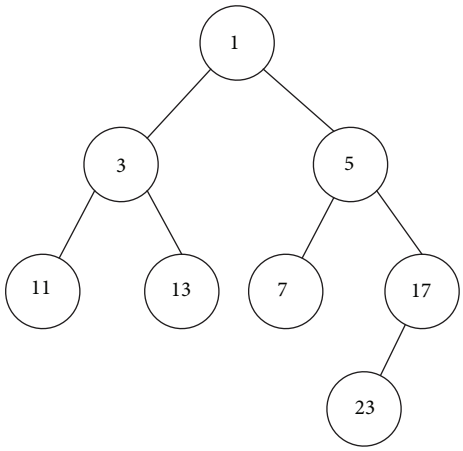


Figure 5: An example of prime-numbered addressing.

(line 2). If the LID of the destination is larger than that of the source, the packet is passed down to the children of node C (line 3). According to the construction of the tree and the creation of LID, the node of which the LID can divide that of the destination is on the path leading the packet directly to its destination. Thus, the right child for the next hop can be chosen (lines 4–7). If there is no child that can pass the packet to its destination, the packet is discarded (line 9). When node C is exactly the destination of the packet, it just holds this packet. If the LID of the destination is smaller than that of the source, the packet is passed to the ancestors of node C and it is directly sent to the parent of C (line 15).

D-HiPr is a novel routing technique by using prime numbers as the sensor nodes' 16-bit local address. Its key advantages meet the requirements of the smart wireless home sensor networks well.

(i) *Nontable-Driven*. By taking the advantage of prime number, D-HiPr uses a single LID for packet routing other than traditional routing table. This mechanism helps to save memory of the transmission nodes as well as reduce traffic delays.

(ii) *Energy Saving*. D-HiPr uses a very simple way to compute routing path and packets in smart WHSN can arrive at the receiver faster and more computation saving to save energy of the relay nodes even it is battery powered.

(iii) *Mobility*. D-HiPr needs no routing table, and the nature of the prime numbers makes the address allocation more flexible. Therefore, D-HiPr can support the mobility of nodes better than traditional routing table.

(iv) *End-to-end Communication*. In smart WHSN with D-HiPr, not all the packets need to be sent via the gateway. In real domain, most packets are directly passed between sensor nodes, and it improves the efficiency of smart WHSN to reduce the traffic of the gateway.

(v) *Medium Scalability*. Although the scalability of prime-numbered addressing is normally hundreds, however, it matches the scalability of WHSN well as it is always composed of hundreds of sensors as well.

4. Incorporating WSN Protocols into WHSN

Based on our hierarchical architecture, in the routing protocol layer, we can use D-HiPr or modify existing WSN routing algorithms in WHSN. Our architecture is compatible with different routing protocols. In this section, we modify three typical WSN routing protocols, LEACH, SPIN, and DD. The key of our improvement is to take the advantage of WHSN hierarchical design by mainly using the fixed plug-in nodes instead of mobile nodes to the response for data transportation so that energy consumption bottleneck of mobile nodes can be avoided and the lifetime of the network can be improved.

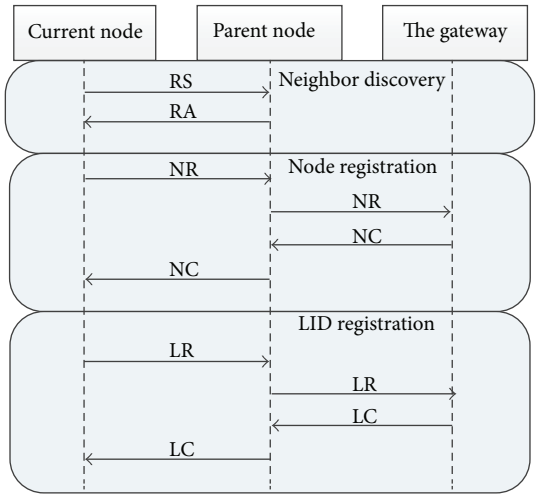


Figure 6: Address allocation process.

```
(1)  for each Node C which holds the packet do
(2)     Comp = packet · Q_d − Q_c;
(3)     if Comp > 0 then
(4)        for i from 1 to n do
(5)           j = Mod(packet · Q_d, Q_{k_i});
(6)           if j = 0 then
(7)              send(k_i, packet);
(8)           else
(9)              discard(packet);
(10)          end if
(11)       end for
(12)    else ifComp = 0 then
(13)       hold(packet);
(14)    else
(15)       send(P, packet);
(16)    end if
(17) end for
```

Algorithm 2: Routing process of D-HiPr.

4.1. LEACH-Pi. Low energy adaptive clustering hierarchy (LEACH) is a low power consumption adaptive clustering routing protocol designed for wireless sensor network. It is based on monolayer clusters and is a data-centric protocol. The key ideal of LAECH is to choose a cluster head randomly in cycle and evenly distribute the global network load to each sensor node. In this way, network energy consumption and the overall survival time of the network can be improved. LEACH routing protocol is composed of two phases, setup phase and ready phase, and each round includes two phases. To reduce protocol overhead, ready phase should be extended longer than setup phase. In the setup phase, each node randomly generates a random value between (0, 1). If it is less than the threshold $T(n)$ based on the next formula, the node will be selected as a cluster head:

$$T(n) = \begin{cases} \dfrac{p}{1 - p * (r \bmod (1/p))} & \text{if } n \in G, \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

As shown in (1), p is the percentage of a node to be selected as the cluster head. r is the round number. G is the group of nodes which are not selected as cluster head in the last $1/p$ round. When the cluster head has been chosen, it broadcasts the message to inform its cluster head position. Nodes decide to join the cluster or not based on the strength of the received message. After a cluster head is chosen, the cluster head will continue to build its cluster. In WHSN, as we explained, node distribution is uneven, and it is unlikely for each node to have an equal chance to be the cluster head. Meanwhile, the WHSN plug-in nodes are more suitable to be a cluster head than randomly chosen node, that is, mobile nodes. It is the key that LEACH protocol should be revised to fit the features of WHSN. By bringing the concept of fixed plug-in nodes in the LEACH protocol, we build the LEACH-Pi where cluster heads are always assigned to plug-in nodes. The process of LEACH-Pi is shown in Algorithm 3.

```
(1)  for each node in WHSN do
(2)     if (node · feature = plug-in) then
(3)        node · state ← head;
(4)        broadcast(Head_Msg);
(5)        Join_Msgs ←receiveJoinMsgs();
(6)        for Join_Msg ∈ Joiln_Msgs do
(7)           node · clusterMb · add(Join_Msg · sender);
(8)        end for
(9)     else
(10)       node · state ← plain;
(11)       Head_Msgs ←receiveHeadMsgs();
(12)       if Head_Msgs ≥ ∅&& state = plain
           then
(13)          sender ← Head_Msgs(0) · sender();
(14)          node · clusterhead ← sender;
(15)          Join_Msg ← "Iam in.";
(16)          send(sender, Join_Msg);
(17)       end if
(18)    end if
(19) end for
```

Algorithm 3: Network building of LEACH-Pi.

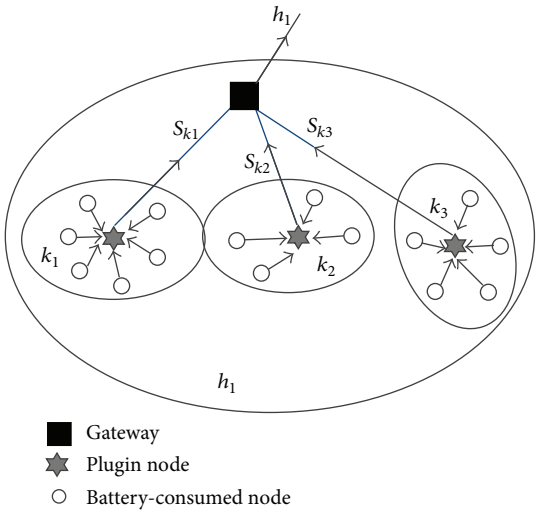


Figure 7: LEACH-Pi.

At first, each plug-in node broadcasts massages to clarify its desire to be cluster header (line 3). Then, it waits for others to join its cluster. After it receives the confirmation messages from its cluster members, the header add them to its cluster member list (lines 6–8). For the nonplug-in nodes, they will receive many header clarifications from cluster headers (line 11). And the first one that sends the clarification will be set to its cluster header (lines 13 and 14). Finally, the non-plug-in node will send a response message ("I am in") to the header (lines 15 and 16). LEACH-Pi routing process is shown in Figure 7.

In the classic LEACH protocol, the most part of energy consumption is to communicate with the cluster heads. In LEACH-Pi, since the plug-in nodes are energy constrained free, their energy consumptions for sending a packet can be neglected. Its energy consumption for communication with cluster heads can be significantly reduced, and the energy consumption of WHSN is merely to communicate between the battery-consumed nodes and cluster heads. Comparing with LEACH, our improved protocol by incorporating the WHSN feature can notably enhance the lifetime of the sensor network bottlenecked by power.

4.2. SPIN-Pi. Sensor protocol for information via negotiation (SPIN) is an adaptive data-centric communication protocol. Its goal is to solve the "implosion" and "overlap" in flooding by using the consultation system and the resource adaptive mechanism between nodes. There are three kinds of data packets in SPIN: ADV, REQ, and DATA. Node uses the ADV to announce that it has data to send, use the REQ to request expects to receive data, and use the DATA to package data [12].

In SPIN routing protocol, as Figure 8 indicates, when a SPIN node obtains new data to share, it broadcasts an ADV request message to clarify its demand. If a neighbor is interested in routing this data, it sends an REQ message as a response, and the DATA is sent to this neighbor node. The SPIN protocol does not take the plug-in nodes into consideration and works well with battery-powered nodes. But in WHSN, there are many plug-in nodes which cannot be ignored and SPIN protocol works with low efficiency.

To improve the routing efficiency and extend the network life, we propose the SPIN-Pi routing protocol based on the original SPIN protocol. As shown in Figure 8, the SPIN-Pi protocol mainly takes advantage of the plug-in node and puts most of the routing tasks on them. In SPIN-Pi routing protocol, for the node that has a packet routing request, the routing process is shown in Algorithm 4. When a SPIN node

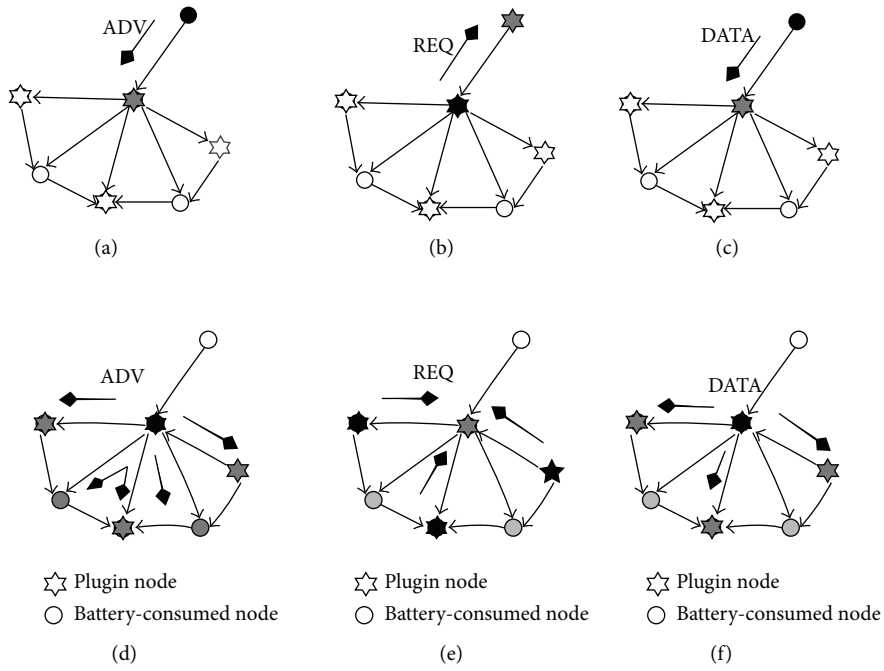


Figure 8: SPIN-Pi.

```
(1)  for each node in WHSN do
(2)     if node · getDATA() ≠ ∅ then
(3)        node · broadcast(ADV);
(4)        REQs ← receiveREQs();
(5)        if REQs ≠ ∅ then
(6)           node · send(REQs · getSend(), DATA);
(7)        else
(8)           node · broadcast(ADV);
(9)           REQs ← receiveREQs();
(10)          if REQs ≠ ∅ then
(11)             node · send(REQs · getSend(), DATA);
(12)          end if
(13)       end if
(14)    end if
(15) end for
```

Algorithm 4: Packet routing request process.

has data to share (line 2), it broadcasts an ADV request message to clarify its desire (line 3). When it receives REQ responses from neighbors (line 4), the packet routing request could be satisfied by these neighbors, and the packet is sent to them (line 6). If there is no one that is willing to route this data, the SPIN node will broadcast an ADV request at the second time (lines 9–11).

For the node that can potentially serve packet routing request, the process in SPIN protocol is shown in Algorithm 5. As the algorithm shows, the plug-in node will serve any ADV request while the non-plug-in node will only route the packet when there is no one willing to serve it in the first request round (lines 5 and 6). Comparing with original SPIN protocol, the energy consumption over the entire network has been significantly reduced by using the plug-in nodes as REQ sender while avoiding mobile nodes involving unnecessary communication.

4.3. DD-Pi. Directed diffusion (DD) is a data-centric routing protocol. The routing process is initiated by the information collection node (the gateway of the WHSN). The information collection node broadcasts *interests* packets to the WSN periodically to tell the network what kind of information it wants to collect. And each node which has received *interests* packet will create a *gradient* for this kind of information. Thus, the gradients for all the information in which the gateway is interested are built in this network. When a piece of information is generated in this network, it will be passed to the gateway according to its gradients.

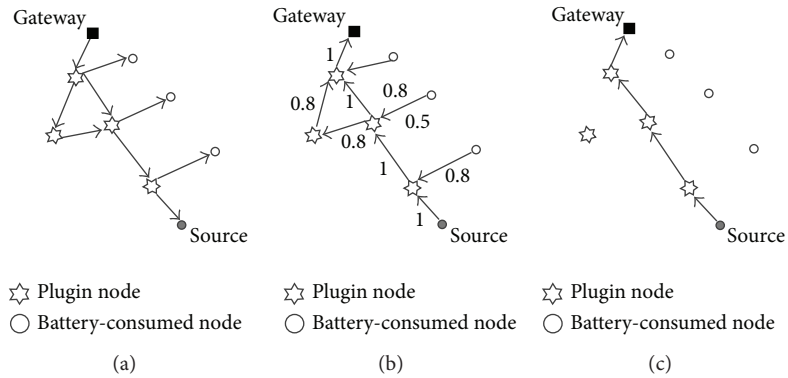


Figure 9: DD-Pi.

```
(1)  for each node in WHSN do
(2)      ADVs ← node · receiveADVs();
(3)      if ADVs ≠ ∅ then
(4)          for ADV ∈ ADVs do
(5)              if (node · feature = plug-in and
                 isFirstRecv(ADV)) or
                 isSecondRecv(ADV) then
(6)                  node · send(ADVs · getSend(), REQ);
(7)              end if
(8)          end for
(9)      end if
(10) end for
```

Algorithm 5: Packet routing response process.

DD protocol is efficient in gathering information for the gateway. To apply this method in WHSN, we propose a modified version of DD protocol called DD-Pi. In this protocol, we define the gradient as the current capability of this node, and the node with a higher capability such as plug-in node has a higher gradient as Figure 9 shows. In this way, most of the communication load can be moved to those plug-in nodes, free of energy consumption bottleneck. Therefore, the network life can be extended.

The DD-Pi routing process is described in Algorithm 6. For each node in WHSN, when an interest message arrives (line 2), the node builds the gradient for this kind of message according to the current capability of the node (line 3). If the current node is a plug-in node (line 4), it will broadcast the interest messages it receives to its neighbors (lines 5 and 6).

5. Network Maintenance

Based on our hierarchical architecture, the path connecting a node to the gateway is unique without any alternative route. Therefore, when any node or link is removed or failed, the network is no longer connected and a fast network maintenance schema responding to node mobilities and failures is imperative to build stable and well-performance wireless home sensor network. Although there have been some path recovery mechanisms used in WSN [13, 14], they may be inefficient and costly according to WHSN architecture and routing protocols. In this section, we introduce mobile node maintenance and path recovery algorithm based on our WHSN architecture.

5.1. Responding to WHSN Mobility. In our wireless home sensor network design, each node is assigned with a prime-numbered address and a unique LID to denote the path to the root. Moreover, the node with mobility is most likely to be connected as a leaf in the tree. When a leaf is moved, we can very easily deal with its motion according to Algorithm 1 that the moved node is deemed as a newly joined node. The only difference is that it has its prime-numbered address other than being assigned with a new one. For example, in a WHSN shown in Figure 10, when a dashed $Node_{19}$ is moved to $Node_{19}$, it will broadcast and rejoin the network with a new parent as root C. Its address will stay as 19 while only its LID changes from 399 to 19 according to its parent's LID. After registering its new LID, it is ready to work at its new position.

5.2. Responding to Backbone Node Failure. Energy exhaustion, hardware problem, or unpredictable moving of backbone nodes can cause network failure and routing paths broken. What is worse, with the hierarchical architecture of the WHSN, all descendant nodes of the failed backbone

```
(1) for each node in WHSN do
(2)    Msgs ← node · receiveInterestMsgs();
(3)    node · buildGradient(node · Cap, Msgs)
(4)    if node · feature = plug-in then
(5)       for Msg ∈ Msgs do
(6)          node · broadcast(Msg);
(7)       end for
(8)    end if
(9) end for
```

Algorithm 6: Routing process of DD-Pi.

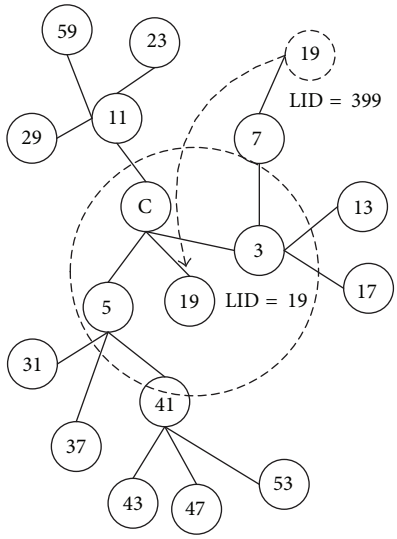


Figure 10: Mobility of D-HiPr.

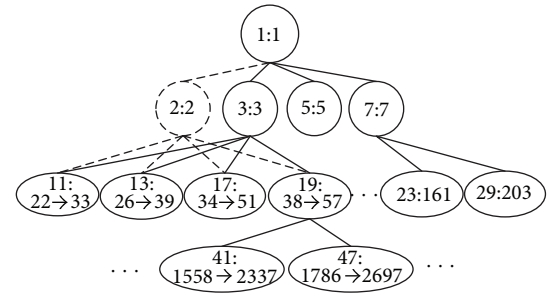


Figure 11: Backbone node failure.

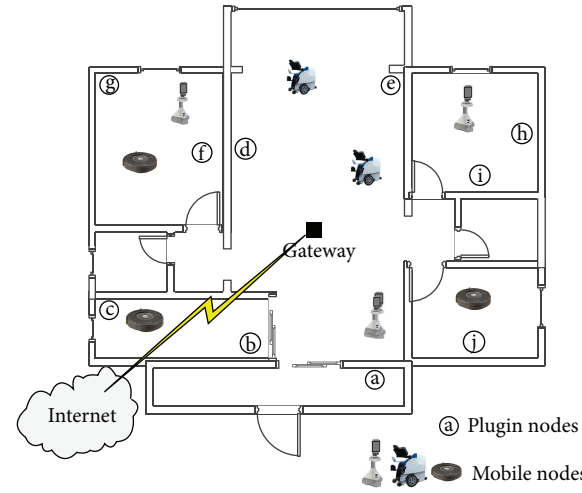


Figure 12: Simulation scenario.

node are disconnected from the network. The cost will be relatively high for all descendant nodes rejoining the network. We propose a mechanism for network restructure that all descendant nodes that are disconnected to their parent node are formed into descendant trees, with the direct child nodes of failed parent node acting as their roots (lines 1-5). Each root of new descendant trees will scan for available parent nodes and select an optimal one from its neighbors (lines 2), while all its descendant nodes keep their associations. Each descendant node gains a new LID from its original parent without sending rejoining messages (lines 7). The gateway will be notified to modify the LIDs for all changes (lines 8). The process for parent node of the descendant trees recovering from failures is described in Algorithm 7.

As presented in Figure 11, each node has a prime address and LID separated by a colon. For instance, $Node_2$ gets failed with its descendant nodes parted into 4 descendant trees: {11}, {13}, {17}, and {19, 41, 47}. Dashed lines denote the links before $Node_2$ gets failed, and arrows in ovals mean the address changes. Node11, Node13, Node17, and Node19 scan for new parent node and resign new LID to their descendant nodes. Each descendant node regains a new LID with inner association links unchanged. This recovery mechanism only involves two procedures. One is that descendant tree root nodes scan for a new parent and the other is that all descendant nodes regain new LID. In contrast with the method that all descendant nodes rejoin the network, it simplifies the network maintenance to avoid all descendant nodes scanning for their parent nodes.

5.3. Simulation and Results. In this section, we present our experiments and results to evaluate our approach. The experiments are based on a simulation testbed of a typical household illustrated in Figure 12. There are plenty

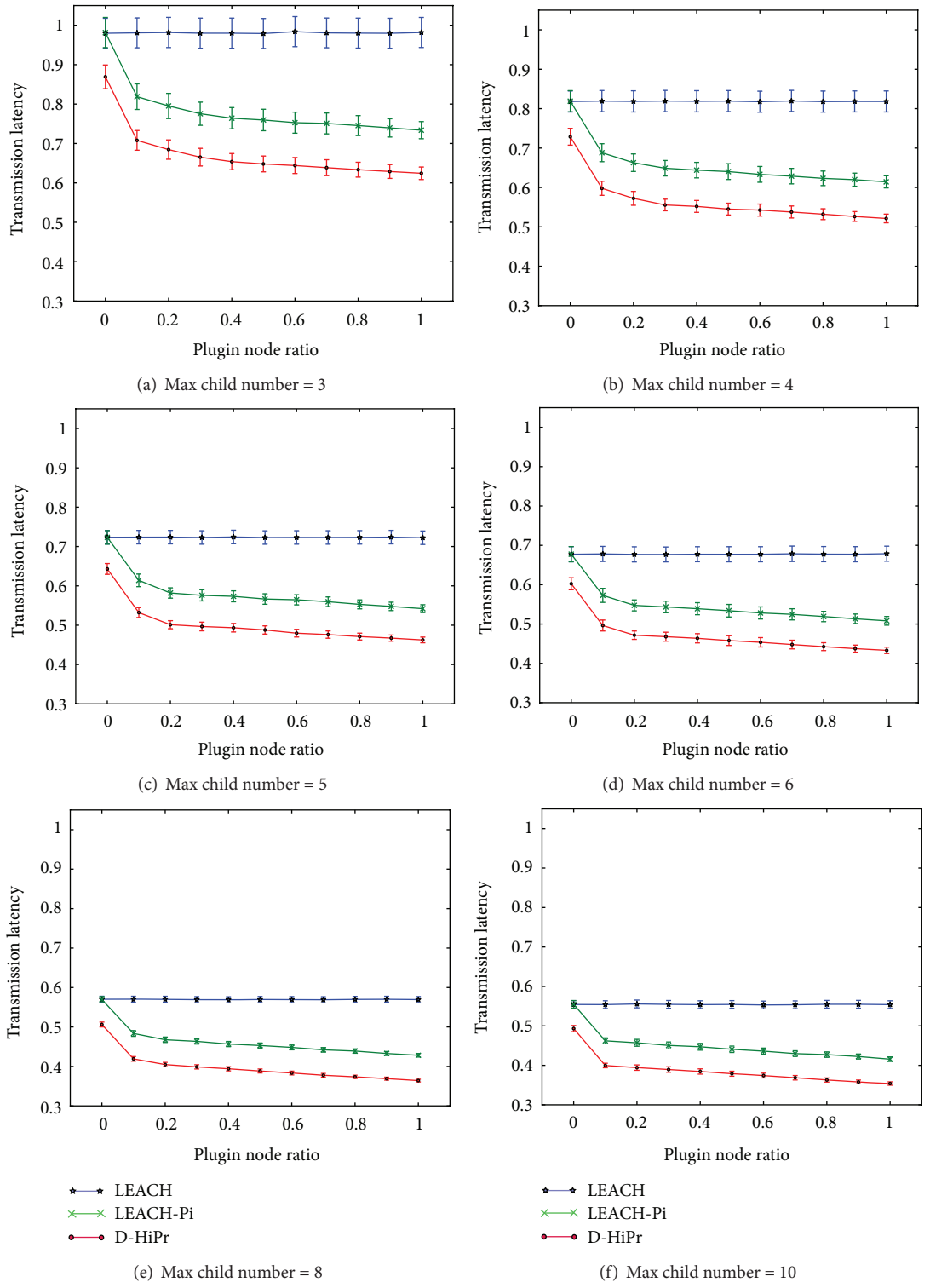
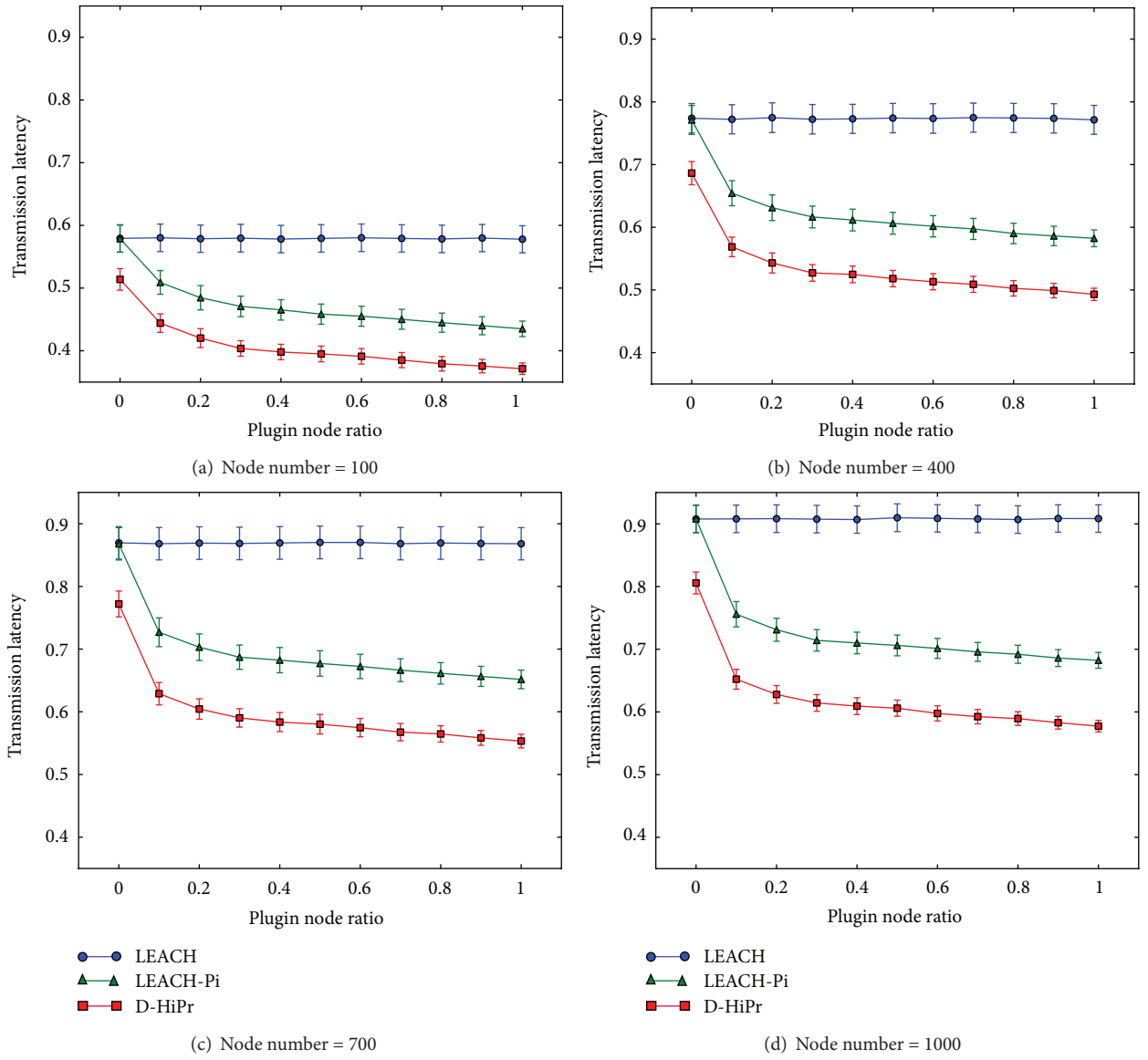


FIGURE 13: Transmission latency by different max child numbers.

of plug-in nodes marked from (a) to (j), such as sensors in the wash machine (a) and in the fridge and the microwave oven in the kitchen (b, c), as well as a bunch of mobile nodes, such as the sensors in household robots and the cleaning machines. Meanwhile, we suppose that the gateway is in the middle of the house. We briefly build three simulations, and in each simulations, based on our hierarchical architecture of WHSN, we testified three algorithms: traditional LEACH, LEACH-Pi, and our proposed D-HiPr algorithm. The simulation results are based on 100 runs.



(a) Node number = 100

(b) Node number = 400

(c) Node number = 700

(d) Node number = 1000

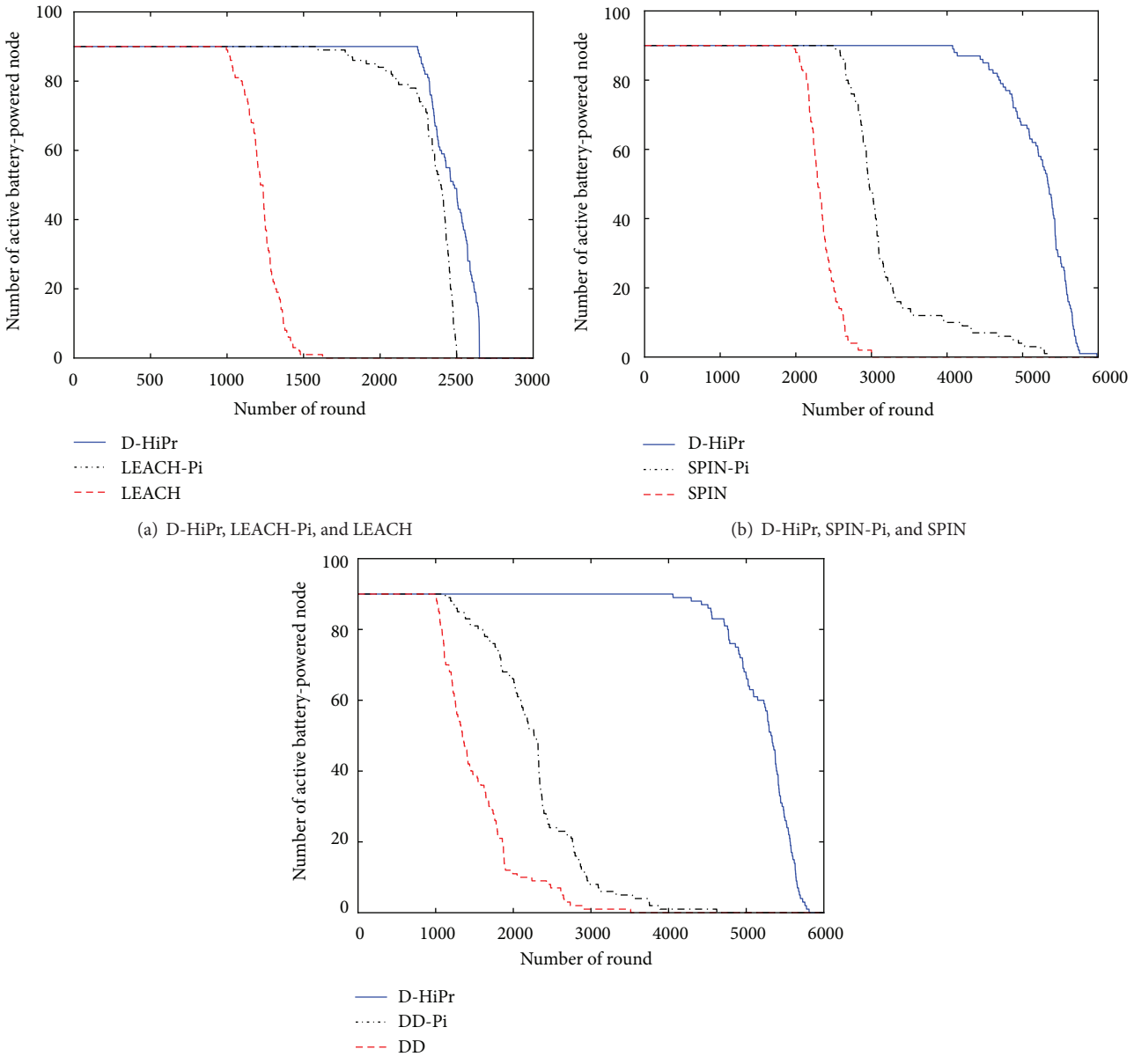
Figure 14: Transmission latencies with different scales of WHSN network.

In the first simulation, we set up a WHSN with 500 of sensor nodes randomly scattered in the household. Although the WHSN always includes two types of sensors, in our experiments, we vary the ratio of plug-in nodes in the network from 0% to 100% as shown in x-axes and testify the average transmission latencies of given 500 packets (shown in y-axes). In this simulation, we have made six experiments, and in each experiment, we set different numbers of maximum allowed children for each node as 3, 4, 5, 6, 8, and 10. The experiment results are illustrated in Figure 13. As we expected, no matter what the settings are, our D-HiPr algorithm, by taking the advantages of hierarchical architecture, plug-in nodes as backbone, and prime-numbered addressing, takes the least transmission latency than the other two algorithms, while LEACH-PI takes the advantage of using plug-in nodes as cluster head which works better than LEACH. We can also see another evidence in the figure that when there are more and more plug-in nodes, D-HiPr and LEACH-PI are benefited with lower transmission latency, while LEACH stays the same. From all six graphs, when the number of maximum children increases and the depth of the tree decreases, all the three algorithms are benefited with lower transmission latency. The reason is that the average distances between nodes are decreased, but all our previous conclusions hold.

In the second simulation, we setup our WHSN with 100, 400, 700, and 1000 sensor nodes, respectively. By varying the ratio of plug-in nodes in the network from 0% to 100% (x-axes), we testify the average transmission latencies (y-axes), which is consistent with the first simulation. From the experiment results shown in Figure 14, our conclusions from the last simulation that D-HiPr takes the most advantages and LEACH-PI takes the advantage of plug-in nodes hold. Moreover, when WHSN scales up, the advantage of D-HiPr and LEACH-PI becomes more and more prominent.

```
(1) for node ∈ failedNode · getChildren() do
(2)     node · parent ← FindNewParent(node);
(3)     node · updateLIDFromParent();
(4)     node · notifyGateway();
(5) end for
(6) for child ∈ node do
(7)     child · updateLIDFromParent();
(8)     node · notifyGateway();
(9) end for
```

Algorithm 7: Path Recovery Algorithm.



(a) D-HiPr, LEACH-Pi, and LEACH

(b) D-HiPr, SPIN-Pi, and SPIN

(c) D-HiPr, DD-Pi, and DD

Figure 15: Network sustain lifetime with different protocols.

In the last simulation, we evaluate our design by comparing the sustain lifetimes of the WHSN when it is applied with different protocols explained in this paper. In this simulation, we set the WHSN with 10 plug-in nodes as shown in the simulation scenario and 90 battery-powered nodes randomly scattered in the house. In each time step, several nodes in the network are chosen to send packets to the gateway. For each packet transmission, if a battery-powered node is involved, it will consume some power for communication. If more battery-powered nodes are involved in packet transmissions, the network will be out of power more quickly. We will record in the simulation how long the battery-powered nodes can keep alive.

The experiment results are briefly shown in Figure 15, and in each graph, we use three different algorithms to be compared. Figure 15(a) shows the sustain lifetime of the network with D-HiPr, LEACH, and LEACH-Pi protocols. As we can see, the LEACH protocol which does not consider the strength of plug-in nodes discharges the battery-powered nodes very quickly and halts the network at about 1300 rounds. In contrast, the LEACH-Pi by taking the advantage of plug-in nodes can survive until more than 2000 rounds, while our D-HiPr protocol by integrating all the features of WHSN works well until more than 2300 rounds.

In Figure 15(b), we use the protocols of D-HiPr, SPIN, and SPIN-Pi to test the network, while in Figure 15(c), we use the D-HiPr, DD, and DD-Pi protocols. Similar to the conclusion drawn from Figure 15(a), because the SPIN and DD do not consider the plug-in nodes, they perform very poorly. SPIN-Pi and DD-Pi by considering the merit of plug-in nodes, they perform better. D-HiPr by taking the advantages of plug-in nodes as backbone and prime-numbered routing algorithm, it performs best, and as Figure 15(c) shows, its sustained lifetime can be triple as that of the DD protocol.

6. Conclusion and Future Works

In this paper, we have designed a hierarchical architecture for smart WHSN. By classifying sensor nodes into two categories, we can take the advantages of using fixed plug-in nodes as backbone nodes for data transportation while the mobile nodes as leaves to keep the flexibility of the network. Based on our architecture design, we have made three major contributions: a novel prime-numbered hierarchical address allocation and routing protocol; an improvement of classic routing protocols for smart WHSN as well as improved network maintenance algorithms to model node mobility and backbone failure. Our simulation results manifest the efficiency and feasibility of our design. However, even if we are capable of dealing with some of the challenges of the domain, we leave many of the others for the future work. Firstly, our design should be integrated with the physical layer controlling protocol such as 802.15.4, but it is not tested. Second, our design should be verified in some real domains applications of WHSN, which will be an imperative work for the next step. Thirdly, connecting with Internet and matching our design with IPv6 network will be good issue left to the future work.

Acknowledgments

This research has been sponsored in part by the National Natural Science Foundation of China no. 60905042, National Key Technology R&D Program of China no. 2012BAI22B05, and Huawei Technology Foundation no. YBNW2010086.

References

[1] Z. S. Wu, "Convergence framework of internet and WSN," Tech. Rep., 2011.

[2] H. Dohler, "Routing requirements for urban low-power and lossy networks," RFC 5488, 2009.

[3] A. Azim and M. M. Islam, "Hybrid LEACH: a relay node based low energy adaptive clustering hierarchy for wireless sensor networks," in *Proceedings of the IEEE 9th Malaysia International Conference on Communications*, December 2009.

[4] W. B. Heinzelman, P. Anantha, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *IEEE Transactions on Wireless Communications*, vol. 1, no. 4, pp. 660–670, 2002.

[5] V. D. Park and M. S. Corson, "A highly adaptive distributed routing algorithm for mobile wireless networks," in *Proceedings of the 16th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '97)*, pp. 1405–1413, April 1997.

[6] L. H. Zhang, L. P. Zhang, and R. H. Huang, "Energy efficient passive clustering based directed diffusion protocol," in *Proceedings of the International Conference on Advanced Measurement and Test*, 2010.

[7] N. M. White and J. E. Brignell, "Sensors in adaptronics," in *UNSPECIFIED Adaptronics and Smart Structures*, Springer, Berlin, Germany, 2007.

[8] J. Li, Y. Guo, and G. Poulton, "Critical damage reporting in intelligent sensor networks," in *Proceedings of the 17th Australian Joint Conference on Artificial Intelligence*, pp. 26–38, December 2004.

[9] Y. X. Li and X. J. Huang, "The simulation of independent rayleigh faders," *IEEE Transactions on Communications*, vol. 50, no. 9, pp. 1503–1514, 2002.

[10] Z. Shelby, S. Chakravarti, and E. Nordmark, "Neighbor discoveryoptimization for low-power and lossy networks," Tech. Rep., June 2010.

[11] J. P. Vasseur and A. Dunkels, *Interconnection Smart Objects with IP: The Next Internet*, Elsevier, New York, NY, USA, 2010.

[12] M. Esler, J. Hightower, T. Anderson, and G. Borriello, "Next century challenges: data-centric networking for invisible computing," in *Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM '97)*, The Protolano Project at the University of Washington, 2000.

[13] G. K. Ee, C. K. Ng, N. K. Noordin, and B. M. Ali, "Path recovery mechanism in 6LoWPAN routing," in *Proceedings of the International Conference on Computer and Communication Engineering (ICCCE '10)*, May 2010.

[14] K. Akkaya and M. Younis, "A survey on routing protocols for wireless sensor networks," *Journal of Ad Hoc Networks*, vol. 3, no. 3, pp. 325–349, 2005.