A Novel Addressing Architecture for Wireless Sensor Networks †

Trong Thua Huynh,
Department of Computer Engineering,
Kyung Hee University, Korea
htthua@networking.khu.ac.kr

Choong Seon Hong
Department of Computer Engineering,
Kyung Hee University, Korea
cshong@khu.ac.kr

Abstract

In this paper, we propose a novel hierarchical architecture for a large wireless sensor network (WSN) wherein sensors are arranged into a multi-layer architecture with the nodes at each layer interconnected as a de Bruijn graph and provide a hierarchical routing algorithm in the network. Using our approach, every sensor obtains a unique node identifier addressed by binary addressing fashion. We show that our algorithm has reasonable fault-tolerance, admits simple and decentralized routing, and offers easy extensibility. We also present simulation results showing the average delay, success data delivery radio in our approach. And we received acceptable results for some potential applications.

1. Introduction

Wireless sensor networks are composed of a large number of sensors densely deployed in inhospitable physical environments. Dissemination information throughout such a network that requires fault-tolerance is a challenge.

Although there are applications as described in [4], [9] which do not require addressing of sensors, we have argued many scenarios where addressing nodes is very essential. However, given that an addresses scheme needs to be build, IP-based addressing to this problem would not be a good solution. For one, IP-based addresses are global unique addresses but WSNs require local unique identifier. In our architecture, sensor nodes within a certain area interact with themselves in a distributed manner and come up with an addressing scheme in which each node obtains a unique local address.

Motivated by above issues and the advantages of the de Bruijn graph admitting simple routing and possessing good fault tolerant capabilities in many interconnected networks, we introduce a multi-layer architecture wherein sensor nodes are interconnected as de Bruijn graph at each layer.

To show how our algorithm supports simple routing, fault tolerance and scalability we introduce a hierarchical routing algorithm that is able to route packets along all possible paths between any pair of the source and destination nodes without performing explicit route discovery, repair computation or maintaining explicit state information about available paths at the nodes.

The remainder of the paper is organized as follows. In section 2, we present the related work. The network architecture is shown in section 3. Section 4 provides routing algorithm for the multi-layer network. We present the performance evaluation in section 5. Finally, we conclude the paper in section 6.

2. Related Work

Many protocols have been proposed for WSNs in the last few years. In works addressed in [1], [3], [4], all sensors have been treated to be alike and are assumed to have similar functionality. In contrast, sensors in our architecture are heterogeneous.

Data dissemination protocols are proposed for WSNs in [2], [3]. SPIN [3] attempts to reduce the cost of flooding data, assuming that the network is source-centric. Directed diffusion [2], on the other hand, selects the most efficient paths to forward requests and replies on, assuming that the network is data-centric. This approach, in company with works addressed in [1], [3], use a powerful concept of data-centric networking for sensor applications. Though this model is very interesting, it may not be applicable to many sensor applications. Certain applications like parking lot networks may require addressability for every sensor node and a method for routing packet to specific nodes. Our architecture can

† This work was supported by a grant No R05-2003-12193-0 from Korea Science and Engineering Foundation.

support both data-centric networks and non data-centric networks.

Clustering algorithms have also been proposed in many literatures, such as GAF [6]. SPAN [7]...etc. The remarkable one among them is LEACH [8]. LEACH is an application-specific data dissemination protocol that uses clustering to prolong the network lifetime. However, LEACH assumes that all nodes have long-range transmission capability. This limits the capacity of protocol and application. Moreover, cluster head failure is also the problem in this approach. In contrast, our approach makes no assumptions like this and does not organize network as clustering architecture. Instead of this, our approach distributes sensors into multi-layers. Each layer is organized as a de Bruijn graph.

3. Network Architecture

In this section, we give a detailed description of the network architecture. In this architecture, we assume that the sensors are immobile. This assumption is reliable especially for indoor applications such as smart home or smart building applications.

Assume that we deploy a network with N sensors. The network architecture can be organized under a hierarchical model which consists of several layers with the node at each layer interconnected as a de Bruijn graph. For a detailed discussion on features of de Bruijn graph, see the paper by Sanmatham et al. [5].Our architecture features is addressed as follows:

- Sensors are organized in a multi-layer architecture with the order of layer numbered increasingly starting from 1.
- kth layer has 2^k sensor nodes.
- Sensors are addressed with binary address form, sensors in kth layer use k bits for node addressing.
- Each node at kth layer is connected to two children nodes at (k+1)th layer and is connected to its parent node at (k-1)th layer (k>1)
- Nodes in 1st layer and 2nd are connected completely. It means that, each node in the first layer has only one neighbor and 2 children. Each node in the second layer has 3 neighbors and 2 children nodes.
- Using graph theoretic notation, the de Bruijn graph BG(d,k) has $N = d^k$ nodes with diameter k and degree 2d. We are interested in binary de Bruijn graph BG(2,k) which have $N = 2^k$. A node x addressed $x_{k-1}x_{k-2} \ldots x_1x_0$ in kth layer (k>2) has 4 neighbors as follows:

$$\text{neig}_1(x) = x_{k-2} \ldots x_1x_0x_{k-1}$$
$$\text{neig}_2(x) = x_{k-2} \ldots x_1x_0\overline{x}_{k-1}$$
$$\text{neig}_3(x) = x_0x_{k-1} \ldots x_2x_1$$
$$\text{neig}_4(x) = \overline{x}_0x_{k-1} \ldots x_2x_1$$

where : $\overline{x}$ is complement of x.

The address of a node in kth layer consists of two parts. One is k bit derived from (k-1)th layer. The other is 1 bit (0 or 1) added from right side. Node x addressed $x_{k-1}x_{k-2} \ldots x_1x_0$ has two children nodes and one parent node. These children are addressed as add($x_{k-1}x_{k-2} \ldots x_1x_0$,0) and add($x_{k-1}x_{k-2} \ldots x_1x_0$,1) while the parent is addressed as rmv($x_{k-1}x_{k-2} \ldots x_1x_0$). The following definitions describe two address transforming functions addition (add) and remove (rmv).

Let K be a k-bit number and y be a binary number. Then

add(K,y) = Ky ; rmv($x_{k-1}x_{k-2} \ldots x_1x_0$) = $x_{k-1}x_{k-2} \ldots x_1$
For example, add(001,1) = 0011; rmv(0110) = 011.

Figure 1 shows an example of the 3-layer network architecture followed by above mentioned features.

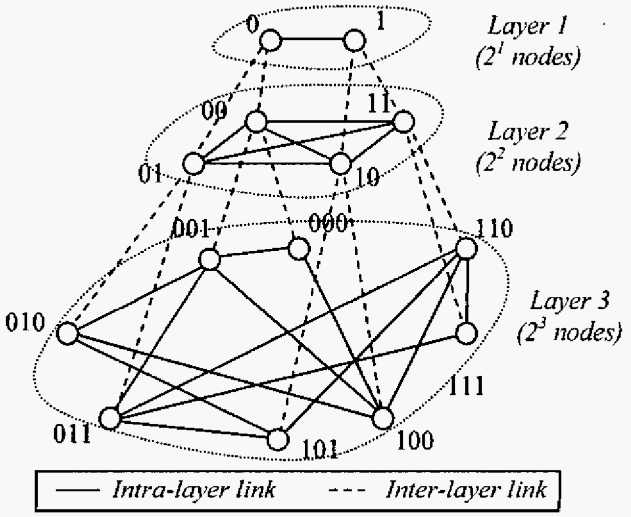

Fig. 1. A 3-layer hierarchical architecture. Node 01 (layer 2) is parent of two nodes 010 and 011 (layer 3) but is one of two children of node 0 (layer 1).

Obviously, the network extension is not issue in this architecture. Whenever sensors need to be deployed into the network, they will be added at the highest layer of the hierarchical architecture. Thus extending network requires a fixed number of interconnections between the new nodes and the nodes at the last layer. If number of new nodes are more than that of the last layer can support, the remaining nodes will make up a new layer being last layer in new network architecture.

4. Routing Algorithm

In this section, we show that packets can be routed throughout the hierarchical architecture. We first consider routing within each layer and then consider routing across layers. To evaluate the routing complexity, we assume that a packet takes unit time to traverse a link.

4.1. Intra-Layer Routing

Routing in the first layer and second layer takes unit time step since the nodes are completely connected. In this section, we describe a simple routing algorithm which is based on the construction of the Bruijn graph.

Let a binary de Bruijn graph have $N = 2^k$ nodes and let $S = s_{k-1}s_{k-2} \dots s_1s_0$ be the source node that sends a packet to the destination node $D = d_{k-1}d_{k-2} \dots d_1d_0$. The packet consists of the data and packet header. The packet header contains the routing information. The packet format is depicted in figure 2.

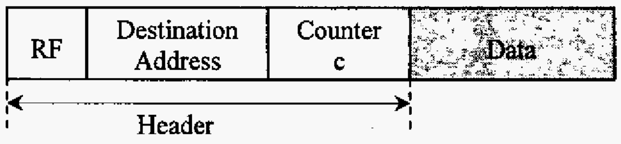

Fig. 2. Packet format. contains the destination node addresses, a counter (c), and a routing flag (RF).

The RF is a 2-bit binary number set to

- 00: the source and destination nodes are in same layer and the source node use Path 1 to route packet to the destination. (default)

- 01: it is the same as 00 but Path 2 is used instead of Path 1. Path 1 and Path 2 will be addressed later.

- 10: the source and destination nodes are in different layers. The source node is at a higher layer than the destination.

- 11: the source node is at a lower layer than the destination.

The counter c is used to record the number of packet hops from the source node to the current node. In addition, it is also used to generate the address of the next node in the path.

Now, we describe the simple routing algorithm in each layer. From the construction of the de Bruijn graph, we know that the source node at kth layer has the following neighbors - $d_0s_{k-1}s_{k-2} \dots s_1$ and $s_{k-2} \dots s_1s_0d_{k-1}$. Using this property we can now generate two paths by appending successive bits of the destination node to the source address.

Path 1

(c=0) $s_{k-1}s_{k-2} \dots s_1s_0$ (source address)
(c=1) $d_0s_{k-1}s_{k-2} \dots s_1$
(c=2) $d_1d_0s_{k-1} \dots s_2$
.
.
(c=k) $d_{k-1}d_{k-2} \dots d_1d_0$ (destination address)

Path 2

(c=0) $s_{k-1}s_{k-2} \dots s_1s_0$ (source address)
(c=1) $s_{k-2}s_{k-3} \dots s_1s_0d_{k-1}$
(c=2) $s_{k-3} \dots s_0d_{k-1}d_{k-2}$
.
.
(c=k) $d_{k-1}d_{k-2} \dots d_1d_0$ (destination address)

Let $x_{k-1}x_{k-2} \dots x_1x_0$ be the address of the node x. The figure 3 describes steps executed by x.

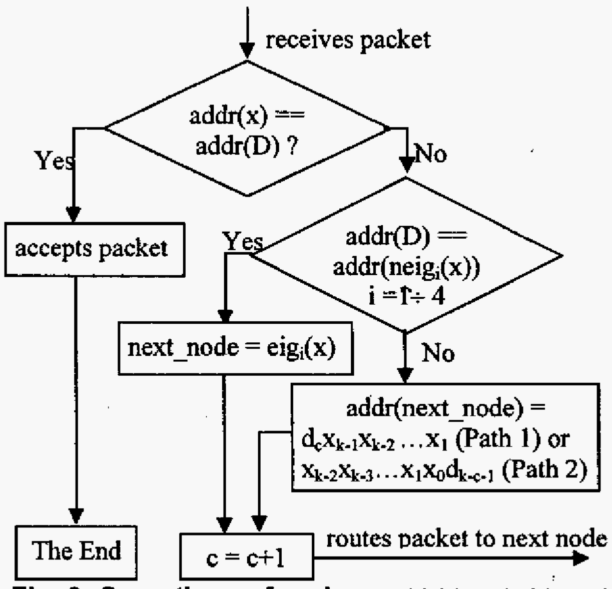
receives packet
addr(x) == addr(D) ?
Yes
No
accepts packet
addr(D) == addr(neig_i(x)) i = 1 ÷ 4
Yes
No
next_node = eig_i(x)
addr(next_node) = d_c x_{k-1} x_{k-2} ... x_1 (Path 1) or x_{k-2} x_{k-3} ... x_1 x_0 d_{k-c-1} (Path 2)
The End
c = c+1
routes packet to next node

Fig. 3. Operations of node x. addr(x), neig$_i$(x), and D present address, ith neighbor of node x, and destination respectively.

4.2. Inter-Layer Routing

In this session, we suppose that the source and destination nodes are in different layers. So, the RF is set to "10" or "11". In addition, the counter c is not incremented in order to maintain a proper value of the counter for intra-layer routing following the inter-layer routing.

In case that the source node is at a higher layer than the destination, the source node first routes the packet to its parent rmv(S). This procedure is repeated recursively until the packet is received by a node at the same layer as the destination node. The RF is set to 00 or 01 now, and the source address is replaced by the address of the node that received the packet. The packet can then be routed to the destination using above intra-layer routing algorithm.

In contrast, when the source node is at a lower layer than the destination, the same procedure is used where the source node first routes the packet to its child using add(S) to generate the address of the next node.

4.3. Fault Tolerant Routing Issue

In a large WSN, it is unrealistic to expect all nodes or links along a path to be free-fault at all times.

Whenever some nodes or links fail, an alternative path that avoids faulty node or link must be derived.

Suppose that nodes x_2 ($d_c x_{k-1} x_{k-2} \ldots x_1$) and x_3 ($x_{k-2} x_{k-3} \ldots x_1 x_0 d_{k-c-1}$) are neighbors of node x_1 ($x_{k-1} x_{k-2} \ldots x_1 x_0$). Also assume that if either node x_2 or the link between x_1 and x_2 has failed, then x_1 chooses the alternative path to node x_3. In addition, x_1 sets RF to 01 (Path 2) and counter c to 0 as well. At worst if both x_2 and x_3 fail, node x_1 routes to one of its two remaining neighbors $d_c x_{k-1} x_{k-2} \ldots x_1$ or $x_{k-1} x_{k-2} \ldots x_1 x_0 d_{k-c-1}$ and sets counter c to 0.

For inter-layer routing, node x_1 chooses another child if the first child fails. In the worst case of both the children failing, node x_1 will route back to one of its neighbors. The approach is the same if parent of x_1 can not be reached. In all cases, counter c must be set to 0 and RF set to default.

5. Performance Evaluation

We developed a simulator based on SENSE simulation [10] to evaluate performance of our approach. The simulation use MAC IEEE 802.11 DCF that SENSE implements. Because network design choices: MAC scheme, network topology, node addressing, and packet routing vary among implementations, we do not compare our nodes to other solutions.

TABLE I:
Simulation Parameters

Parameter	*Value*
Network size	250x250
Number of sensors	254
Packet generating rate	1 packet/sec
Data packet size	64 bytes
Simulation time	1000 sec
Radio of transmission range	20m

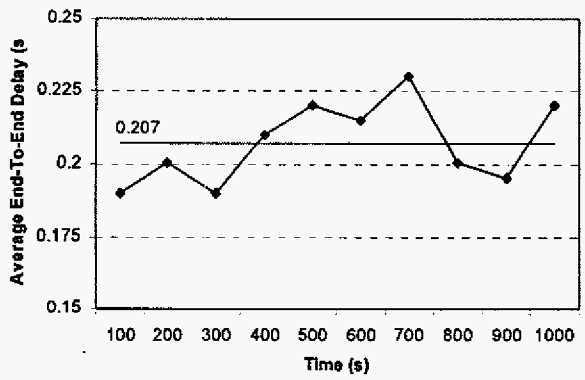

Fig.4. Average end-to-end delay as a function of simulation time

In this simulation, we take account on two metrics: the end-to-end delay and the success data delivery radio with and without node or link failure. The end-to-end delay refers the time taken for a data packet to be generated until the time it arrives at the destination. The success data delivery radio is the rate of the number of successfully received data packets at the destination and the total number of packet generated by a source. Some of simulation parameters are listed in Table I.

To illustrate the performance of our architecture, we choose randomly pairs of source and destination nodes in order to communicate each other. For 254 node network, the simulation results in figure 4 depict that the average end-to-end delay in this network is approximately 0.207 seconds. Obviously, this end-to-end delay is quite low and it is an acceptable value in many potential applications.

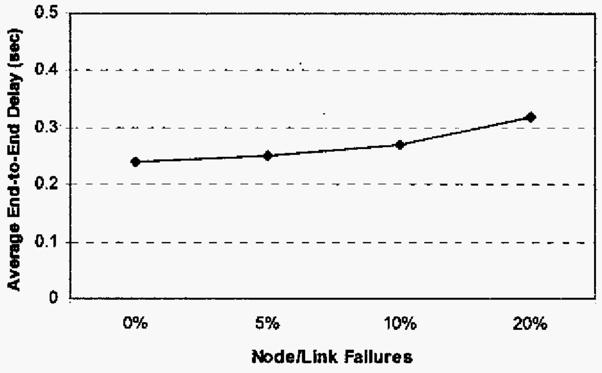

Fig.5 (a). Average end-to-end delay as a function of node/link failures

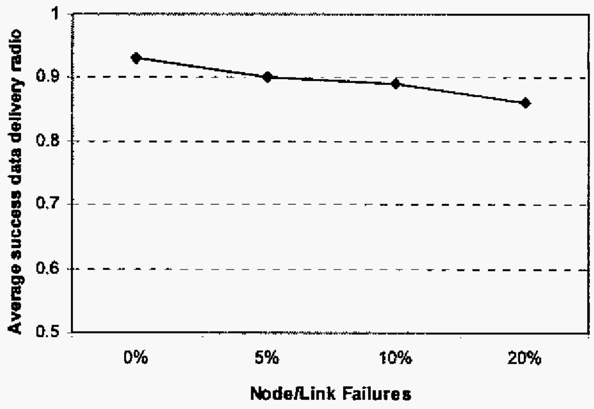

Fig.5 (b). Average success data delivery radio as a function of node/link failures

To indicate the fault-tolerance of our algorithm, we inject failures into the simulated network and evaluate the performance with 5%, 10% and 20% node/link failures one after the other. Results in figure 5(a) depict that although the average end-to-end delay increases when the network size increases, the increase of average delay is not significant (0.24 second delay for none node or link failure, 0.32 second delay for 20% node or link failures).

In figure 5(b), we study the average success data delivery radio metric as a function of node or link failures. And the results indicate that the success data delivery radio is decreased when the number of node or link failures increases. However, this decrease is not significant. And it still ensures this radio at quite high level accepted in a large amount of sensor network applications. When the next node or the link to next node of the node is fail, it is straightforward to the current node alters to another path (Path 1 or Path 2) without performing explicit route discovery, repair computation or maintaining explicit state information about available paths at that node. That is why the increase of average delay and the decrease of success data delivery radio are insignificant when the number of node/link failures increases.

The simplicity and efficiency of our architecture are examined by simulation in figure 6. We study metric average success data release radio as a function of sensor network size. To do this, we generate a variety of sensor fields of different sizes, ranging from 14 to 510 nodes in increments of 2^i (i = 4,5,6,7, and 8). When the number of node is changed, network size is also proportionally changed by scaling the square and keeping the radio range constant in order to approximately keep the average density of sensor nodes constant. Results show that the data is delivered successfully in the network is quite high (always more than 90%).

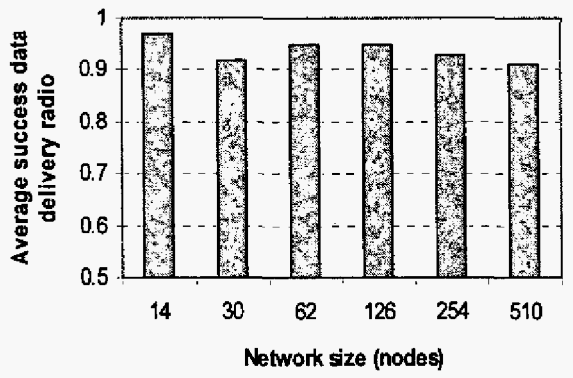

Fig.6. Average success data delivery radio as a function of network sizes

6. Conclusion and Future Work

This paper describes a hierarchical architecture for WSNs wherein sensor nodes are distributed into layers. Nodes in each layer are interconnected as a de Bruijn graph. Our architecture easily solves fault tolerance and extensibility. We also provide a simple routing algorithm between any two nodes in the network.

We created simulations based on SENSE [10] in order to illustrate the performance and bring out the main goal while studying this approach. Because sensors have limited computation, in this paper, we do not investigate into the shortest path algorithm that consumes much energy than above mentioned simple routing algorithm. Nevertheless, since multi-year battery life is preferable for future sensors, we need to future study on optimal routing to improve the goodness our architecture. As one of future works, we need to study an optimal energy scheme in order to save energy for sensors.

References

[1] Wendi R.Heinzelman et al., "Energy efficient communication protocol for wireless microsensor networks", *in Hawaii International Conference on System Sciences, January, 2000.*

[2] C.Intanagonwiwat et.al., "Directed diffusion: A scalable and robust communication paradigm for sensor networks", *in ACM Mobicom 2000.*

[3] J. Kulik, W. Heinzelman, and H. Balakrishnan, "Negotiation-based Protocols for Dissemination Information in Wireless Sensor Networks", *in ACM/IEEE MobiCom '99, Seattle.*

[4] Deborah Estrin et.al., "Next century challenges: Scalable coordination in sensor networks", *in MOBICOM'99, Seattle.*

[5] M. R. Samantham, and D. K. Pradhan, "The de Bruijn Multi-processor Network: A Versatile Parallel Processing and Sorting Network for VLSI", *IEEE Transaction on Computers, April 1989.*

[6] Y. Xu, J. Heidemann, and D. Estrin, "Geography-Informed Energy Conservation for Ad Hoc Routing," *in ACM/IEEE on Mobile Computing and Networking (MOBICOM), Italy, July 2001.*

[7] B. Chen et al., "Span: An Energy-Efficient Coordination Algorithm for Topology Maintenance in Ad Hoc Wireless Networks," *in ACM Wireless Networks, September 2002.*

[8] W. R. Heinzelman et al., "An Application-Specific Protocol Architecture for Wireless Microsensor Networks," *IEEE Transactions on Wireless Communications*, October 2002.

[9] Jeremy Elson and Deborah Estrin, "An address free architecture for dynamic sensor networks", *submitted for publication. http://www.isi.edu/ estrin/papers/*

[10] http://www.cs.rpi.edu/~cheng3/sense/