This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the ICC 2008 proceedings.

Testbed Experimentation of a Meshed Tree Routing with Local Link State for Wireless PAN Mesh

Rui Zhang, Tae Rim Park, Myung J. Lee, Hakyung Jung*, Jaehong Ryu**
Dept of Electrical Engineering, City University of New York, NY, 10031
* ECE Dept, Seoul National University
** Electronic Telecommunications Research Institute (ETRI)
{rzhang, taerim, mjlee}@ee.ccny.cuny.edu, *hkjung@mmlab.snu.ac.kr, **jhryu@etri.re.kr

Abstract*— This paper focus on the testbed experimentation of a meshed tree routing algorithm with local link state for Wireless Personal Area networks (WPANs) based on current IEEE 802.15.4 MAC and PHY. The meshed tree uses a block addressing scheme based on tree structure and local link state information for mesh routing. Our approach exploits the information from the global tree structure for the direction of packet forwarding and local link state for choosing the next hop toward the destination. It has two prominent features: scalability and fault tolerance. Each node maintains a local link state of k-hop(usually 2-hop) information regardless of network size, which makes the approach scalable. Being a mesh, our approach shows good fault tolerance and load balancing. Testbed experiments show that the meshed tree displays superior performance when compared with AODV and a tree based algorithm. Comparisons were made with respect to packet deliver ratio, energy consumption and memory usage. The proposed algorithm is being considered as the routing algorithm for IEEE 802.15.5 WPAN Mesh standard.

I. INTRODUCTION

Wireless sensor networks, especially Wireless Personal Area Networks (WPANs) with the release of IEEE 802.15.4 standard [1], will soon find applications in homes, offices, industrial, and commercial premises. Due to potentially large size networks, it is a challenge to design a scalable robust routing algorithm, considering the sensor node with critically constrained resources in view of battery, memory, and processing. Another difficulty for routing protocol design comes from the lossy and dynamic nature of wireless links, burdening the maintenance for stable routes with low-power radio transceivers.

In this paper, we present the experimental aspect of a novel meshed tree routing algorithm, which integrates a tree topology, each node with k-hop (usually 2-hop) local link state information, and the logical block addressing. The combination of block addressing and global tree structure permit a packet to be guided to the general direction of the destination, and the local link state directs the packet to the next hop.

Compared to various existing routing algorithms such as AODV[2], Directed Diffusion[3], OLSR[4], our approach has two prominent features. First is its scalability. Most routing algorithms need certain message flooding throughout the whole network. Although some hierarchical algorithms use the idea of dominant sets, MPR, clusters or hybrid [5, 6] to avoid pure flooding, the control overhead still increases along with network size for the maintenance of link state table or routing table at each node. In contrast, our meshed tree is a fully distributed algorithm incurring little overhead. Each node broadcasts several Hello messages to its k-hop (default, 2-hop) neighbors during the initial auto-configuration phase, and spends only several hundred bytes of link state information for those neighbors, regardless of the network size.

The second feature is to support multiple paths and load balancing for fault tolerance purpose since relay decision is always based on local link state. The quality of local links is periodically monitored to dynamically adjust the local forwarding decision. All these have been implemented in our test-bed based on IEEE 802.15.4 PHY and MAC standard [1]. We use the Micaz as our base platform, which embeds CC2420 wireless transceiver and Atmel128 microcontroller [7]. The CC2420 transceiver provides the Link Quality Indicator (LQI)[8], which can be used to update the local link states adapting to dynamic wireless environments.

The paper is organized as follows. Section 2 describes the meshed tree algorithm and Section 3 introduces the testbed implementation. Experiments and analyses are presented in Section 4, and conclusions are given in Section 5.

II. MESHED TREE ROUTING ALGORITHM

The routing algorithm has two major components: tree formation and distributed link state generation. The block address is assigned during the network initialization stage and used for routing as well as network auto-configuration. After initialization, hello messages are used to build the mesh links based on basic tree structure.

A. Tree formation and block address assignment

The tree formation starts with the first node in the network designating itself as the root and beginning to accept association requests from other nodes. After a node is successfully associated, it determines as a parent node whether to allow other nodes to join the network through it. In the case of using IEEE 802.15.4 devices as in our testbed, Full Function Devices (FFDs) [1] accepts new nodes as children,

The research is supported by Electronic Telecommunications Research Institute (ETRI).

while Reduced Function Devices (RFDs) cannot accept new nodes. When a joining node receives multiple association responses, it should choose a proper node to join (the details will be discussed in the next section). After the tree reaches its bottom, that is, no more nodes are waiting to join the network (a proper waiting time is to be given by application), all leaf(bottom) nodes will report a desirable number of addresses to their parent nodes including its own address and some additional addresses reserved for further use if it wants. An address management policy is needed to control these additional address requests. Meanwhile, a parent node will calculate the number of requested addresses from its children nodes after all requests are reported, then add its own address need and report the total number of addresses to its parent. This process repeats until the root of the tree receives the information from all the branches, then it begins to assign addresses. Figure 1 shows an example of the address report by node C.

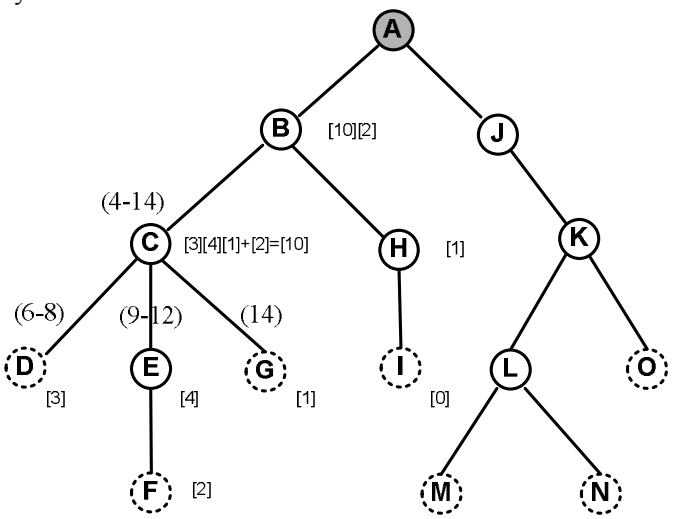

Figure 1. Tree Formation and address assignment. [3] means the number of required addresses is 3, and (6-8) means that the block address is 6 to 8

The address assignment, takes a top-down procedure. First, assuming the total number of requested addresses is less than the total number of addresses available, the root will assign a block of consecutive addresses to each branch below it, taking into account the requested number of addresses. This procedure continues until the bottom of the tree is reached. After address assignment, a tree is formed and each node has an address table for tracking its branches. For example, in Figure 1, the node C has a block address (4-14) because the block addresses for three branches are (6-8), (9-12), (14), respectively. and it adds the address block (4-5) to use the address 4 for itself and the address 5 as reserved for potential expansion of the tree.

B. Local Link State(LLS) scheme

After tree formation, every node builds up its own local link state (LLS) information. The additional complexity of combining the block addressing with the LLS scheme is minimal. A node broadcasts several hello messages to exchange link state information with its immediate neighbors when it receives an address from its parent. The Hello message contains its own block address (begin and end address), tree level, and the block addresses of one hop neighbors. Upon receiving the hello messages, a node constructs a 2-hop neighbor information and wireless mesh links. Figure 2 show the 2-hop neighbors of node J.

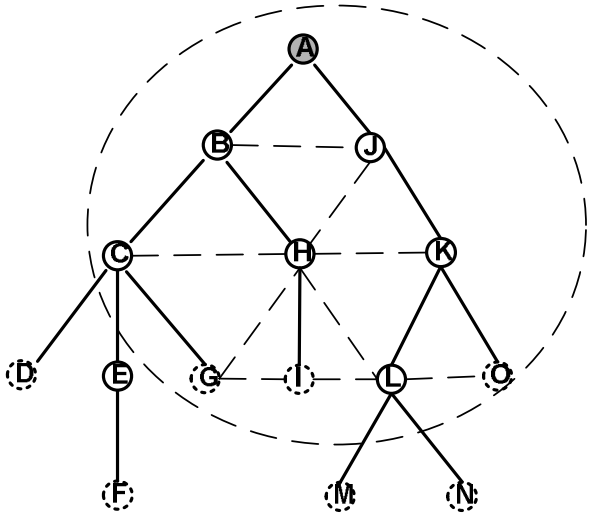

Figure 2: The 2-hop neighbors of node J, the solid lines constitute a tree and dashed lines indicate mesh links. Both links form a meshed tree.

Whenever a node receives the hello message, it will update its link state information. The LLS information contains two parts: neighbor list and connectivity matrix. The neighbor list is shown below in Table 1:

Table 1: Neighbor-List

$begAddr_1$	$endAddr_1$	$treeLevel_1$	$hops_1$	$linkQuality_1$
$begAddr_2$	$endAddr_2$	$treeLevel_2$	$hops_2$	$linkQuality_2$
…	…	…	…	…
$begAddr_N$	$endAddr_N$	$treeLevel_N$	$hops_N$	$linkQuality_N$

$begAddr_i$ is the beginning address of the block address owned by neighbor i; $endAddr_i$ is the ending address of the block address owned by neighbor i; $treeLevel_i$ means the tree level of neighbor i; $hops_i$ is the number of hops from "this node" to neighbor i; *linkQuality* describes the the value of link quality from CC2420. Note that the link quality field is valid for one hop neighbor.

From the one-hop neighbor information included in each hello message, a node can construct a connectivity matrix. We implement the connectivity matrix using bitmap to conserve RAM space. For example, Table 2 illustrates the connectivity matrix for the node J.

The plus('+') and minus('-') at the cross cell of two nodes indicate they are or are not directly connected. The matrix is symmetric because only bi-directional links are recorded. Shown here is the half of the matrix. The matrix will become non-symmetry if we account asymmetric links.

Table 2: Connectivity Matrix for node J in Figure 2

	J	A	B	C	G	H	I	K	L	O
J		+	+	-	-	+	-	+	-	-
A			+	-	-	-	-	-	-	-
B				+	-	+	-	-	-	-
C					+	+	-	-	-	-
G						+	+	-	-	-
H							+	+	+	+
I								-	+	-
K									+	+
L										+
O										

C. Data Forwarding

We present the data forwarding algorithm in Figure 4. Every node will calculate the next hop by its own address and local link state information. Each node checks whether the destination is one of its descendants, one of its neighbors, or their descendants. Otherwise, it will choose a next hop in the direction toward the destination. The ability of our algorithm to suggest the general direction toward the destination is worth mentioning. For instance, even if the destination address is not found in the 2-hop neighbor list, we can select the next hop from one of the neighbor nodes in the list that is closest to the root node. This in general shortens the route to the destination. Currently, the meshed tree algorithm presented here is being considered as the IEEE 802.15.5 WPAN Mesh standard.

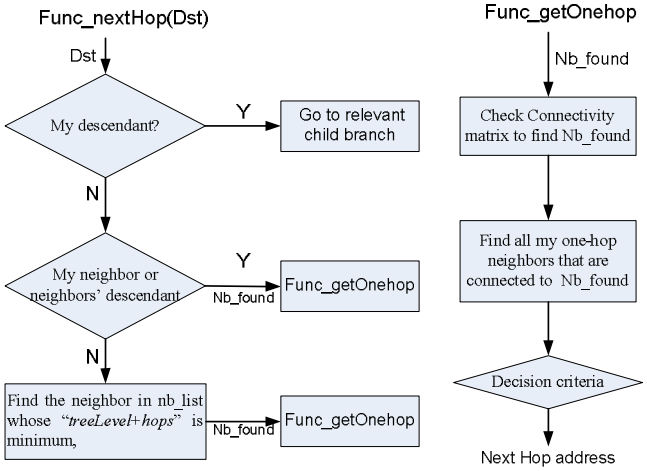

Figure 4. Data forwarding scheme

III. IMPLEMENTATION ON TESTBED

A. Testbed Architecture

A simulation study is reported in [9] to assess the performance of our routing algorithm. To further demonstrate the feasibility of the algorithm in real applications, we implement it in a WPAN mesh testbed. The testbed contains 50 Micaz nodes each connected with a MIB600 gateway [7] deployed in the 5th floor of Engineering building, as shown in Figure 5. For proper radio coverage and redundancy in connectivity, the node deployment needs to consider real signal measurements.

Fig. 6 shows the architecture of our testbed. The User Network is basically Internet running TCP/IP protocols. The user network allows external collaborators test their algorithms using our testbed via the testbed server. Control Network is used to upload Micaz with the binary executable files for algorithms to test. Control messages and testing results between the sensor nodes and the server are also exchanged through the Control Network.

The hardware platform of the Target Network is using Micaz based on IEEE 802.15.4 PHY and MAC stack.. We developed a common software platform to control MAC functions. The implementation of the proposed algorithm is written by C and compiled with AVR-GCC.

B. Implementation

The network starts from a node designating itself as tree root, then other nodes who want to join this tree will first generate passive scan request. Upon receiving this request, all potential parents will respond with their beacon frames, and the joining node, by examining the LQI value indicated in the beacon frames, will choose the one with highest LQI value as the parent to join by using IEEE 802.15.4 association procedure [1]. We use the non-beacon mode to construct the tree. The association procedure needs the message exchange (request and response) between two nodes so that it can construct a tree structure with bi-directional links. After a node successfully joins the network, if the node is FFD, it can accept new nodes as children by responding with beacons when hearing scan request. In our testbed implementation, all nodes are the FFDs.

After the tree is established, the hello message will be exchanged to form a *meshed*-tree. Every time a node receives a hello message from a neighbor node, existing or new one, it checks if the sender of the hello message exists in its one hop neighbor list. If not, it will add source block address of the sender of this hello message to its neighbor list. As mentioned earlier, the hello message contains the sender's one hop neighbor list, which permits the receiver of this hello message to build the two hop neighbor list. If the hello message contains k-hop neighbor list, then, each node can build (k+1)-hop neighbor list. The link quality in the neighbor list is the result of the moving average of LQI [10,11]. Note that here the link quality computation can use different algorithms such as ETX or ETT [12]. For the data forwarding, the decision criteria in Func_getonehop in Figure 4 depends on application

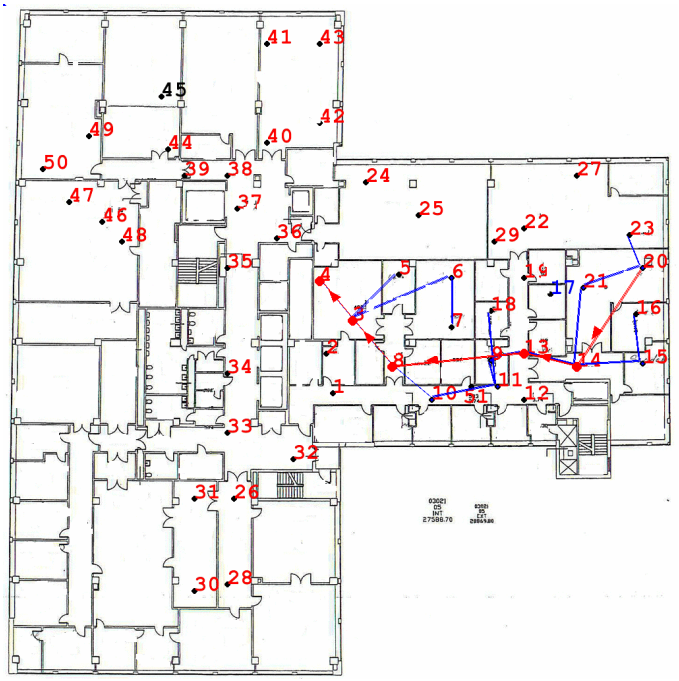

Figure 5. Testbed deployment and an example topology: the blue lines represent tree structure and red lines represent the routing path.

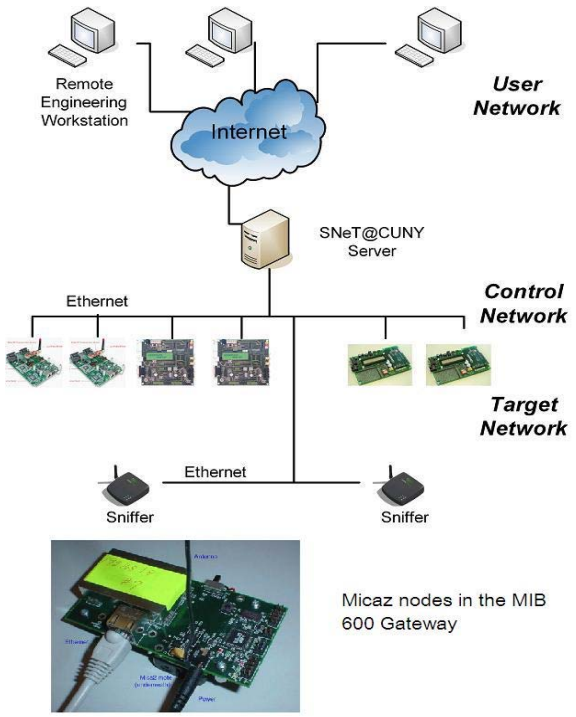

Figure 6: User network scenario and sensor board

or user's policy. In our implementation we use LQI to choose next hop node toward the destination, which supports the better packet deliver ratio [12,14]. Other criteria may include battery capacity, direction to the destination, utilization, security, etc.

IV. EXPERIMENAL RESULTS AND ANALYSES

For performance comparison, we implemented three algorithms on the testbed: the proposed meshed-tree routing algorithm, a tree based routing on IEEE 802.15.4 [15], and AODV. Several performance metrics are evaluated. First we examine the packet deliver ratio. The experiment starts when the node 4 in Figure 5 designates itself as the tree root. Other nodes will join the network until the tree level gets 8, meaning the data from farthest nodes will need 8 tree hops to arrive at node 4. In our experiments, 6 groups of node pairs, each with a source and a destination, are selected with respect to the number of hops ranging from 2 to 7. For each group we randomly choose 5 pairs of nodes. For each pair, 48 bytes-long packets are generated from the source at the rate of one packet per second suggested in [16], and the duration of each trial is 100 seconds. The first experiment is for the packet delivery ratio, which is defined by the total number of packets received by the destination node divided by the total number of packets generated from the source node. For each hop, the packet delivery ratio takes the average of 5 node pairs. We perform this experiment twice creating two different tree topologies. Aforementioned, the first experiment starts the tree from the node 4, while for the second experiment, the tree started from the node 20. Figure 7 depicts the packet deliver ratio versus the hop distance. Note here that the hop distance means the number of hops routing along tree structure.

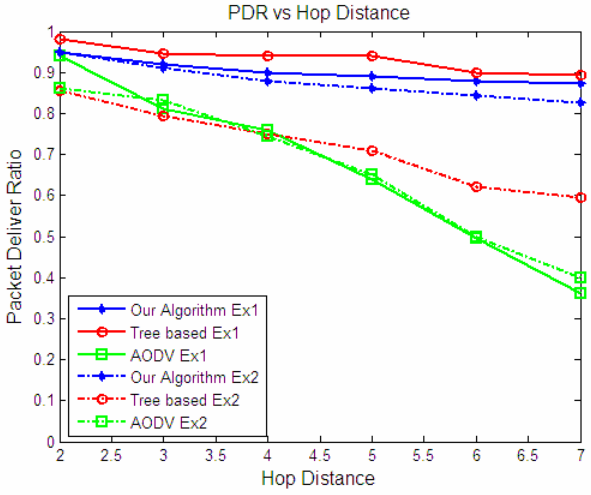

Figure 7: Performance result: hop distance vs PDR

From this figure we can see that AODV performs poorly. By examining experimental results, we find several reasons contributing to AODV's poor performance. First, the network wide flooding causes packet collision between RREP and RREQ especially when the network size gets large, which often prevents the route between a source and a destination cannot be established. Second, even after a route established, data from the source sometimes to be delivered to the

destination. This affirms that the shortest path found by AODV makes the distance between the two adjacent nodes large, which in turn, makes the signal power received at the next hop toward the destination may not be enough for correct packet reception. Also we observe that the performance of tree based routing is very sensitive to the tree topology created. If the tree is constructed with high quality and stable links, the packet delivery ratio is high, otherwise the performance worsens. Our routing algorithm maintains high packet deliver ratio compared to other two algorithms and, importantly, the effect from tree topology change appears negligible.

In the second experiment, we consider the end-to-end data delivery cost for the routing algorithms. Initially we are interested in the comparison for control overhead as many do, however, it would be a unfair comparison because of the nature of algorithms experimented. Tree-based algorithms, essentially belonging to "semi" proactive category, incurs control overhead for network initialization and maintenance via hello messages, while the same for reactive routing AODV incurs whenever a new data demands a new route. Therefore, we choose to experiment data delivery cost instead, ignoring the overheads associated with network initialization and route discovery. We define the end-to-end delivery cost as the total number of MAC transmissions incurred by successfully delivering a packet from a source to a destination, accounting all the transmissions including retransmissions at each hop along the route. Note that IEEE 802.15.4 MAC tries up to 4 retransmissions for a unicast frame before informing the upper layer of the link failure. The data delivery cost is determined by two factors: the hop count and the number of retransmissions at each hop. First, the larger the hop count, the more the end-to-end delivery cost. Second, to cope with the lossy nature of wireless links, the MAC layer adopts frame retransmission, which could also increase the total transmission times.

We maintain the same experiment scenario as in the first. For every node pair, we repeat the experiment twice, each one lasting 50s. We do not count lost packets. Figure 8 depict the histogram of end-to-end data delivery cost for three approaches. The horizontal axis represents the total number of transmissions and the vertical axis for its occurrence during the experiments.

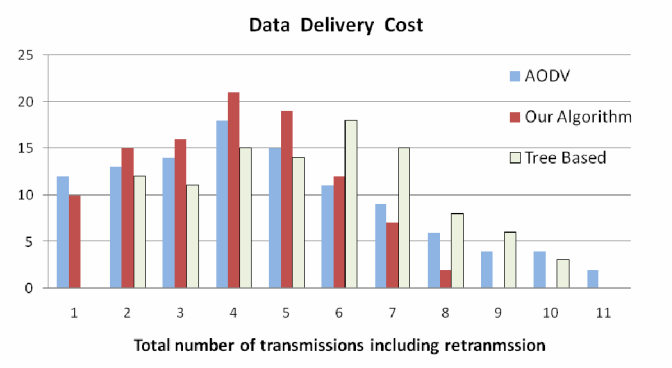

Figure 8. Histogram of the total number of transmissions

As observed from the figure, tree based routing suffers from excessive retransmission, much more than AODV and our algorithm. There are two major reasons: first, tree based routing always takes much more hops than AODV and our algorithm. The second is its sensitivity to topology change, as exhibited in the first experiment. For almost all cases, AODV performs in the middle. As explained earlier, the quality of the links derived from AODV may be poor, causing many MAC level retransmissions. This aggravates the performance especially when the route distance tends to be large, as shown by the right-end-tail of the figure. Our Meshed tree is the winner of this contest, primarily thanks to the use of LQI when forming the meshed tree.

The final experiment is about the fault-tolerant ability. The links in WPAN networks are fragile and can be dropped down by many reasons such as the exhaustion of batteries or wireless environment changes. Therefore, the tolerance from link failures is a very critical issue for the seamless operation. To address this issue our experiment considers the following. The node 20 has a temperature sensor and reports its reading to the sink node 4 in Figure 3. The distance between these two nodes is about 45m. After confirming that the temperature reading has been successfully reported, we power-off a node randomly selected from 21 nodes, node IDs 3 to 23 except node ID 4 because it is the sink node. We examine whether the data can still be delivered to the sink. We repeat this trial 100 times for each routing algorithm tested and measure the percentage the temperature data is still reported to the sink. Figure 9 depicts the results.

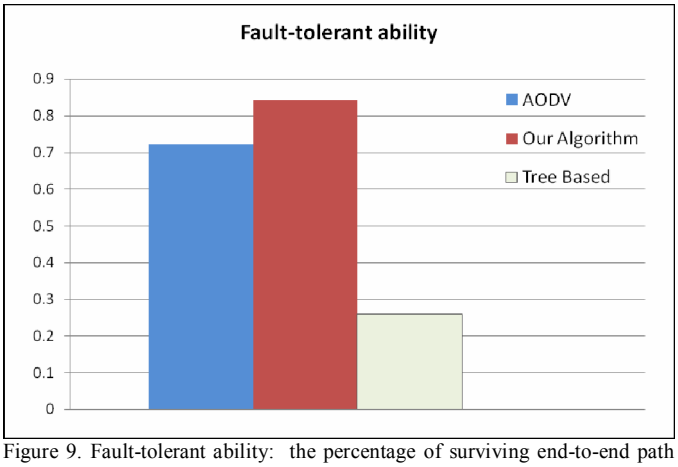

Figure 9. Fault-tolerant ability: the percentage of surviving end-to-end path when a certain node in network is failed

It again demonstrates that our meshed-tree perform much better than the other two routing algorithms due to its 2-hop LLS information maintained at each node. If a route fails, the upstream node of the faulty node or link can use alternative routing paths by examining its LLS table. The tree routing is extremely vulnerable to link or node failure, because a node failure in the tree path will disrupt packet transmissions via the faulty link or node. Also, the tree repair procedure is very complex especially when the tree topology is associated with underlying addressing scheme as in the tree algorithm for this

experiment [17], and it explains why such tree repair is rarely reported in the literatures for the wireless mesh networks. In AODV, if the original path fails, it will re-initialize the RREQ to find a new path. The performance degradation of AODV is explained before.

V. CONCLUSIONS

Routing algorithm design in Wireless Personal Area Networks (WPANs) is challenging because of the stringent resource constraints such as battery, processing, and memory. In this paper we present a meshed-tree routing protocol based on tree structure with local link state information at each node. It uses block addressing to avoid address shortage and hello messages to build up the wireless mesh links and update link state information. The routing is done by collaborating the tree based block address and local link state information. By having local link state information, the proposed meshed-tree avoids single point of failure problem inherent in tree-based approach and enhances scalability.

We implemented a tree-based routing, AODV, and our meshed-tree on a 50-node testbed deployed in a floor of our Engineering building. Several experiments have been performed to evaluate the algorithms. In all these experiments, our routing algorithm demonstrates desirable performances in packet deliver ratio, routing cost, and fault-tolerant ability.

The future works include how to use the link quality or some network statistics to improve network performance. The proposed routing algorithm is being considered as the mesh routing algorithm for IEEE 802.15.5 WPAN Mesh standard.

REFERENCE

[1] "Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LRWPANs)," IEEE 802.15.4 Standard, 2006.

[2] C. E. Perkins, E. M. Belding-Royer and S. Das, "Ad Hoc On Demand Distance Vector (AODV) Routing," *IETF Internet draft*, draft-ietf-manet-aodv-13.txt, Feb. 2003.

[3] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks," *Proc. ACM Mobicom'00, pp. 56-67.*

[4] A.Qayyum, L.Viennot. and A.Laouiti. "Multipoint relaying: An efficient technique for flooding in mobile wireless networks", *Technical Report RR-3898, INRIA,2000*

[5] IF.Akyildiz, etc, "Wireless sensor networks: a survey", Computer network, Volume 38, Issue 4, 2002

[6] Z. J. Haas and M. R. Pearlman, "The zone routing protocol for ad hoc networks," IETF: draft-ietfmanet-zone-zrp-02.txt, Jun. 1999.

[7] Crossbow MICAz Mote Specifications. [Online]. Available: http://www.xbow.com

[8] CC2420 datasheet :http://www.chipcon.com

[9] J.Zheng, and M.J.Lee. "A resource-efficient and scalable wireless mesh routing protocol", *Special Issue of Ad Hoc Networks Journal on Wireless Mesh Networks,* Sept. 2006, Elsevier.

[10] A.Woo, T.Tong, and D.E.Culler, "Taming the underlying challenges of reliable Multihop Routing in Sensor Network", *In Proceedings of the ACM Sensy 2003,* CA, USA.

[11] D. Lal and et al. "Measurement and Characterization of Link Quality Metrics in Energy Constrained Wireless Sensor Networks", in *Proc. Of IEEE GLOBECOM 2003*, San Francisco, USA, December 2003.

[12] D. De Couto and et al. "A High-Throughput Path Metric for Multi-Hop Wireless Routing", *In Proceedings of the ACM Mobicom 2003,* San Diego, CA, USA

[13] J. Zhao and R. Govindan, "Understanding Packet Delivery Performance in Dense Wireless Sensor Networks," in *Proc. of ACM SENSYS 2003*, CA, USA, November 2003.

[14] D. Son and et al, "Experimental study of the effects of Transmission Power Control and Blacklisting in Wireless Sensor Network," in *Proc. of IEEE SECON*, Santa Clara, CA, USA, 2004.

[15] F. Cuomo and et al. "Routing in Zigbee: benefits from exploiting the IEEE 802.15.4 association tree", *In Proceedings of the IEEE ICC 2007,* Glasgow, Scotland.

[16] J.Zheng, and M.J.Lee. "Will IEEE 802.15.4 make ubiquitous networking a reality?:A discussion on a potential low power, low bit rate standard", ,IEEE Communication Magazine, June 2004.