LIEMRO: A Low-Interference Energy-Efficient Multipath Routing Protocol for Improving QoS in Event-Based Wireless Sensor Networks

Marjan Radi, Behnam Dezfouli, S.A.Razak, K.A.Bakar
Faculty of Computer Science and Information Systems
Universiti Teknologi Malaysia (UTM)
Johor, Malaysia
radi@ieee.org, dezfouli@ieee.org, shukorar@utm.my, knizam@utm.my

Abstract — **In the recent years, multipath routing techniques are recognized as an effective approach to improve QoS in Wireless Sensor Networks (WSNs). However, in most of the previously proposed protocols either the effects of inter-path interference are ignored, or establishing low-interference paths is very costly. In this paper, we propose a Low-Interference Energy-efficient Multipath ROuting protocol (LIEMRO) for WSNs. This protocol is mainly designed to improve packet delivery ratio, lifetime, and latency, through discovering multiple interference-minimized node-disjoint paths between source node and sink node. In addition, LIEMRO includes a load balancing algorithm to distribute source node's traffic over multiple paths based on the relative quality of each path. Simulation results show that using LIEMRO in high traffic load conditions can increase data reception rate and network lifetime even more than 1.5x compared with single path routing approach, while end-to-end latency reduces significantly. Accordingly, LIEMRO is a multipath solution for event-driven applications in which lifetime, reliability, and latency are of great importance.**

Keywords – *Wireless Sensor Networks; QoS; Multipath Routing; Load Balancing; Interference.*

I. INTRODUCTION

According to the special characteristics of WSNs (such as limited energy resources, low bandwidth, and unreliable links), designing appropriate routing protocols to satisfy application-specific QoS demands is very importance. With respect to the advantages of multipath routing techniques (such as elevated reliability, higher throughput, improved load balancing, and lower delay), compared with single path approaches, this method has been widely used as an effective approach to improve QoS in different networks [1][2][3]. Meanwhile, due to unique features of wireless channel in low power WSNs, benefits of multipath routing seem to be less obvious than wired networks [4][5]. Accordingly, in order to design an effective load balancing technique for WSNs, we must consider special characteristics of wireless communication (such as inter-node interference) in conjunction with multipath routing protocols.

Current research on multipath routing technique can be divided into three main categories, based on the primary criterion used in protocol design. In the first group, multipath routing is essentially used to improve fault tolerance [6][7][8][9]. These protocols utilize one path for data transmission and may switch to another path if a failure or significant degradation of resources occurs in the currently active path. Although this method prevents network disjointedness, yet, it is costly and time consuming. In addition, since only one path is used at a time, congestion may be a problem.

In the second group [10][11][12], multiple copies of data packets are sent through different paths towards the sink. Consequently, these protocols demonstrate higher resilience against path failure and provide higher reliability. Their disadvantage is that they suffer from the heavy overloads caused by transmitting multiple copies of data packets over several paths.

In the last group, multipath routing protocols are intended to increase network lifetime via balancing energy consumption throughout the network [13][14]. To this aim, these protocols utilize load-balancing algorithms to distribute traffic over multiple paths. Although, energy related issues are considered in protocol design, unfortunately, wireless link properties and their effects on performance are ignored. Therefore, load balancing algorithms cannot provide significant improvement in network throughput. This issue is mainly referred to as "route coupling" and severely limits the performance of multipath routing protocols.

Motivated by the disadvantages of each category, we propose a multipath routing protocol (LIEMRO) to improve packet delivery ratio, lifetime, and latency, which are the important QoS parameters of event-driven applications in WSNs. The proposed routing protocol consists of a multipath routing protocol and a load balancing algorithm. In this approach, a set of node-disjoint interference-minimized paths are established from the source to the sink, while these paths impose minimum interference over each other (to minimized route coupling effect). Moreover, extra routes are only established if they don't decrease data reception rate at the sink node.

The rest of the paper is organized as follows: In Section II we introduce route establishment and maintenance in LIEMRO. In Section 0 we introduce the proposed load balancing algorithm for LIEMRO. Performance evaluation is performed in Section 0. We conclude and provide directions for future works in Section V.

II. Multi path routing Protocol

In this section, we introduce detailed operations of path establishment and maintenance in LIEMRO.

A. Initialization Phase

At the initialization phase, each node obtains some information regarding its neighbors. At the first step of this phase, each node broadcasts a fixed number of control packets and records the number of successfully received packets from its neighbors. Therefore, a preliminary neighbor table, which also contains link qualities (corresponding to forward and backward packet reception rates) can be formed in each node. In this phase, we use ETX metric [15] as our link cost function. In the second step, sink node sets its cost to zero and broadcasts this cost to its neighbors. When a node receives this packet, it retrieves the contained cost and saves that as the accumulated ETX cost of the neighbor node to the sink. Then, it updates the cost contained in the packet by adding this cost to the link cost of the node from which this packet has been received. Afterward, the new cost is broadcasted. For example, when node J receives a broadcast packet from node I, it adds the cost contained in the broadcast packet to the cost of link J-I. This process is repeated until all nods calculate their cost to the sink node. At the end of this step, each node has obtained the cost of packet transmission to the sink node through its neighbors. During the normal network operation, broadcast frequency of ETX values depends on network dynamics (for example, when a neighbor node along the path to the sink dies, the new cost must be broadcasted).

B. Route Discovery and Establishment Phase

This phase is initialized when a set of sensor nodes detect an event, and after some data aggregation, they select a node as the source node to send the aggregated data to the sink node.

To start route establishment process, a route request packet (*Route_request*) is sent from the source to the sink. At each node, the best next hop neighbor is selected according to the following cost function:

$$Cost_{i,j} = (ETX_j + \frac{1}{p_{i,j}q_{i,j}}).(\frac{1}{ResBatt_j}).(1 + Interference\_Level_j) \quad (1)$$

In (1), ETX_j is the ETX cost of node j to the sink, contained in the neighbor table of node i. $p_{i,j}$ and $q_{i,j}$ are the forward and backward packet reception rates between node i and node j, respectively. $ResBatt_j$ is the remaining battery of node j and $Interference\_Level_j$ is the maximum interference level that node j has experienced. Calculation of $Interference\_Level_j$ value will be explained later.

In addition to forwarding *Route_request* packet, each node also keeps the ID of the node from which this packet has been received. At each node, upon reception of *Route_request* packet, a variable named *Route_path* is set to

1 to denote that this node is selected to be a part of a route[1]. When the *Route_request* packet reaches to the sink node, sink node replies to this packet by transmitting a *Route_reply* packet to the node from which *Route_request* packet has been received. Along the reverse path from the sink to the source, receiving *Route_reply* packet at each node forces it to set its *Route_path* variable to *2*. This means that a confirmed path passes through this node. Meanwhile, when the *Route_reply* packet moves from the sink to the source, some nodes may overhear this packet. In order to estimate the overhearing level experienced by these nodes, backward packet reception rate can be used. For example, when node *A* overhears a *Route_reply* packet from node *B*, node *A* refers to its neighbor table and adds value $q_{A,B}$ to its *Interference_Level* variable and then broadcasts this value. Therefore, neighbor nodes will be aware of maximum interference level that node *A* has experienced. Upon receiving *Route_reply* packet at the source node, the first path is formed and is ready to be used. Concurrently with data transmission over the first path, creation of the second route can be started by the source node. This is performed by sending the second *Route_request* packet to the sink. To establish a new route, each node's best next hop is the neighbor node which has minimum cost (according to (1)) and is not part of any route to the sink (i.e., its *Route_path* variable is equal to *0*). Similarly, when the second *Route_request* packet reaches to the sink, sink node sends the *Route_reply* packet along the reverse path to the source. Overhearing of *Route_reply* packet by any node, forces that node to update its interference level. This is performed by adding the new interference level to the *Interference_Level* variable. When the second route is formed (i.e., *Route_reply* packet reaches to the source node), source node's traffic can be distributed over the first and second paths, using load balancing algorithm (the proposed load distribution algorithm will be discussed in Section 0).

After creation of the second path, no effort will be made to establish the third path until a feedback message is received from the sink node. In this algorithm we use Received Packets Throughput (RPT) parameter to decide if it is suitable to create another path or not. RPT is defined as the total number of received packets at the sink in a time unit. When the *n-1*th path is created, sink node can compare the RPT of *n-2* paths to the RPT of *n-1* paths to decide about the creation of *n*th path. If receiving data packets from *n-1* paths results in higher RPT than receiving data packets from *n-2* paths, a positive feedback will be sent from the sink to the source to indicate next path can be established. No special packet is defined to transmit feedback message; instead, a special field is added to the ACK packets. Therefore, source node will receive sink node's feedback after multi-hop propagation of ACK packets. If the received feedback is positive, creation of the next path will be started, if it is negative, *n-1*th path (i.e., the last created path) will be disabled. As long as the source node receives positive

---

[1] At the initialization phase, *Route_path* and *Interference_Level* are initialized to zero to indicate that no path passes through this node and no interference has been experienced.

feedback for newly created paths, a similar process will be performed to establish another path to the sink. This process will be terminated when a negative feedback is received.

To facilitate protocol understandability and in order to provide more implementation details, respective pseudo codes of source node, intermediate nodes, and sink node are shown in Pseudo Code 1, Pseudo Code 2, and Pseudo Code 3, respectively.

C. Route Maintenance

Although route failure is less probable in event-driven applications, compared with monitoring applications [2], nevertheless, we have considered the situation in which a node on an active path fails to deliver its data packets to the next hop node.

Considering S-MAC [16] as the MAC layer protocol, if any node on an active path, after k efforts, could not receive CTS packet or ACK packet from the next hop node, then it sends an error message to the source node (through the reverse path). After the source node receives such error massage, it disables the path from which this message has been received and redistributes network traffic over other active paths. In addition, to prevent performance degradation, source node initiates a new route discovery process.

Assuming at least two efforts, variable k can be calculated using geometric distribution:

$$k = \frac{1}{p_{i,j}} + 1 \qquad (2)$$

Pseudo Code 1. Source node's algorithm.

1.	**If** (no *Route_request* packet is sent before)
2.	{
3.	**For** (all of this node's neighbors)
4.	/*if no path passes through neighbor node i*/
5.	**If** (*Route_path$_i$*==0)
6.	Calculate *cost$_i$* for neighbor *i*;
7.	Send *Route_request* to the node which has minimum *cost$_i$*;
8.	}
9.	**If** (*Route_reply* for *n*th path is received)
10.	Transmit data packets over the *n* created paths, using load balancing algorithm;
11.	**If** (a positive feedback is received for *n*th path)
12.	{
13.	Continue data packet transmission over the *n* created paths, using load balancing algorithm;
14.	**For** (all of this node's neighbors)
15.	**If** (*Route_path$_i$*==0)
16.	Calculate *cost$_i$* for neighbor *i*;
17.	Send *Route_request* to the node which has minimum *cost$_i$*;
18.	}

2 Because in event-driven applications data packets will be sent when an event occurs, accordingly, the period of data transmission and thereby the chance of node failure is lower than the case in which data packets are being sent frequently from all the nodes to the sink.

19.	**If** (a negative feedback is received for *n*th path)
20.	Disable the *n*th path;
21.	Transmit data packets over the *n-1* previously created paths, using load balancing algorithm;

Pseudo Code 2. Intermediate nodes' algorithm.

1.	**If** (*Route_request* packet is received)
2.	{
3.	**For** (all of this node's neighbors)
4.	**If** (*Route_path$_i$*==0)
5.	Calculate *cost$_i$* for neighbor *i*;
6.	Send *Route_request* to the node which has minimum *cost$_i$*;
7.	*Route_path*=1;
8.	}
9.	**If** (*Route_reply* packet is received)
10.	Send this packet in the reverse path to the source node ;
11.	*Route_path*=2;
12.	**If** (*Route_reply* packet is overheard from node *i*)
13.	{
14.	Refer to neighbor table and extract backward packet reception rate to node *i*;
15.	Add the extracted value to *Interference_Level*;
16.	}

Pseudo Code 3. Sink node's algorithm.

1.	**If** (the first *Route_request* packet is received)
2.	{
3.	Send *Route_reply* packet in reverse path;
4.	}
5.	**If** (the *n*th *Route_request* packet is received)
6.	{
7.	Calculate RPT of using *n-1* paths;
8.	**If** ((RPT of *n-2* paths)<=(RPT of *n-1* paths))
9.	Send positive feedback;
10.	**Else**
11.	Send negative feedback;
12.	}

III. LOAD BALANCING

For better utilization of each path's resources, in this section, we propose a load balancing algorithm in conjunction with LIEMRO.

A. Prerequisites of Load Balancing

Whenever a *Route_reply* packet is received at the source node, it means that a new route is established and source node can transmit a portion of its data packets through this path. In addition, since *Route_reply* packet moves from sink to source, it can contain some information regarding its traversed path. This information can be useful to apply an efficient load balancing algorithm.

The higher the experienced interference level, the longer the latency of the path (due to some issues such as higher

contention for medium access and more packet corruption rate), consequently, less traffic must be issued to higher interference paths. In addition, in order to improve network lifetime, energy consumption must be balanced throughout the network. It means that paths with lower residual battery must transmit lower data rate to save their remaining energy. Therefore, to run a proper load balancing algorithm at source node, average interference level, average residual battery, and estimated ETX value of each path must be known[3]. Note that these values must be updated regularly by each node. For example, consider the situation in which source node has just received the *Route_reply* packet of the third path. At this time, source node can distribute its traffic over all three paths, but the load balancing algorithm may be working on inaccurate values of each path's interference level; because, when the third path's *Route_reply* packet travels from the sink to the source, nodes belonging to the first and second path may experience new levels of interference. Therefore, along every path, each node adds the newly perceived interference level to its *Interference_Level* variable. Afterward, ACK packets are used to propagate these updated values. A similar operation is applied according to the residual battery and ETX values whenever they are changed more than 10% compared to their last broadcast values.

B. Load Balancing Mechanism

Suppose source node receives an ACK packet from node j on its kth path, it can compute *load cost* of path k as:

$$p_k = (ETX_j + \frac{1}{p_{src,j}q_{src,j}}) + (1 + AccIntfr_k) + (\frac{1}{AccResBatt_k}) \qquad (3)$$

ETX_j is the ETX value at node j and $p_{src,j}$ and $q_{src,j}$ are forward and backward packet reception rates between source node and node j. $AccIntfr_k$ and $AccResBatt_k$ are the accumulated interference level and accumulated residual battery from sink to node j, respectively.

The optimal data rate ratio (r_k) of path k can be obtained as follows:

$$min(r_1p_1 = r_2p_2 = r_3p_3 = ... = r_np_n)$$

$$\qquad (4)$$

Subject to:

$$\sum_{k=1}^{n} r_k = 1$$

Therefore:

$$r_k = \frac{1}{p_k \sum_{f=1}^{n} \frac{1}{p_f}} \qquad (5)$$

---

[3] The later one is intuitive: the path with higher ETX value incurs lower packet corruption rate and lower energy consumption.

Radio			
Modulation	FSK	Encoding	NRZ
Output Power	0 dBm	Frame	45 bytes
Transmission Medium			
Path Loss Exponent	4	$PL_{D\,0}$	55 dBm
Noise Floor	-105 dBm	D_0	1 m
MAC Protocol (S-MAC)			
Sleep Duration	115 ms	Wake Duration	500 ms
Contention Window for SYNC	15 slots	Contention Window for Data	31 slots
Other Parameters			
Number of Nodes	200	Area	100×100 m^2
Deployement	random	Simulation Time	600 sec
Battery Capacity	2400 mAh	Queue Length	4 KB

IV. PERFORMANCE EVALUATION

We used OMNeT++ framework to program our simulation scenarios. The simulation parameters are demonstrated in

Table 1. These parameters are based on the characteristics of Mica 2 motes and data link model of [17].

Sink node is positioned at the up right corner of the area. In addition, to maximize the path between the event detection area and sink node, we placed source node at the down left corner of the field.

As we mentioned earlier, LIEMRO tries to construct node-disjoint paths for each detected event. Nevertheless, paths with shared nodes are probable when two or more events occur in the network. Therefore, we have also evaluated LIEMRO in multiple-event situation. To this aim, an additional source node is located at the center of the area to maximize path interferences.

In the rest of this section, we evaluate LIEMRO under various packet generation intervals and transmission power levels. Note that for all single path approaches, ETX is used as the route selection metric. In each figure, Transmission Power is abbreviated as TP.

A. Data Reception Rate

Data reception rate is defined as the ratio of successfully received packets at the sink node to the total number of packets sent from the source. As it can be seen from Figure 1(a), at high traffic condition, LIEMRO can increase data reception rate up to nearly 2x compared with single path approach. In fact, through creating multiple interference-minimized paths, and by applying quality-based traffic distribution over them, average packet queue length per node will be decreased and the amount of packet loss due to congestion will be reduced.

Figure 1(a) shows that the data reception rate through three paths is less than the data reception rate through two
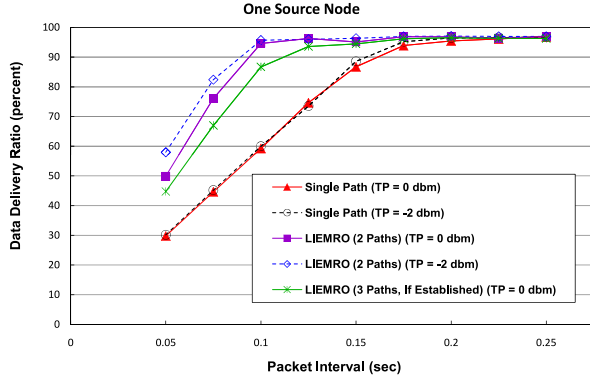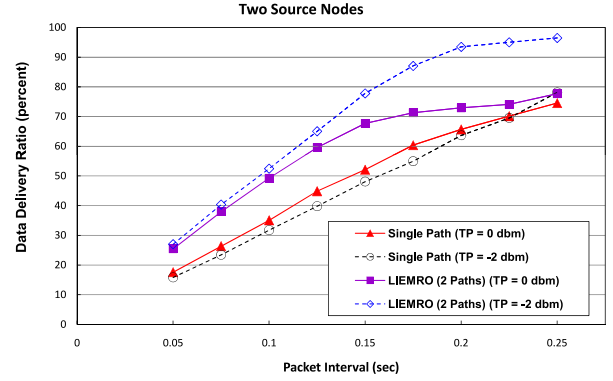
Figure 1. Data delivery ratio of single path routing and LIEMRO versus different traffic loads.

paths. This can be explained as the effect of third path on the first and second paths. In this scenario, creation of the third path imposes higher inter-path interference, which results in lower data reception rate at the sink node. Accordingly, the third path is disabled and source node's traffic is sent over the first and second paths.

Figure 1(b) indicates that even if multiple events occur in the network, LIEMRO still surpasses single path approach. However, as we expected, paths with shared nodes cause higher inter-path interference and more severe contention for medium access, which result in lower data reception rate.

In both Figure 1(a) and Figure 1(b), with similar number of paths, lower transmission power provides higher data reception rate. This is mainly due to the less inter-path interference of lower transmission power.

B. Network Lifetime

To measure the efficacy of proposed load balancing algorithm, lifetime is defined as the period of time from network initialization to the time first node dies.

According to Figure 2(a), transmitting data over two paths results in higher lifetime due to more balanced energy consumption in the network. However, at high traffic loads, LIEMRO and single path routing both provide very low lifetime. Generally, from the MAC layer point of view, contention for medium access and listening duration must be increased as the network traffic intensifies. Since we have considered S-MAC as the medium access protocol, this condition eventuates in its higher duty cycle, which ultimately results in lower network lifetime. Additionally, higher contention for medium access outcomes in more retransmissions (i.e., longer transmission duration per packet) as a result of increased packet corruption rate. From the network layer viewpoint, in heavy traffic conditions, each path of LIEMRO is used at its saturated level; therefore, network lifetime based on the death of first node will be almost the same for single path and multipath approaches.

If we consider transmission power level equal to 0dBm and compare the lifetime of *LIEMRO (2 Paths)* with

LIEMRO (3 Paths), we observe that using three paths outcomes in higher lifetime. This is can be explained as the effect of more balanced energy consumption when three paths are in use. However, as explained in Section 0-A, when the third path is established, data reception rate at the sink node is reduced and LIEMRO disables this path. Accordingly, even though network lifetime may be increased by establishing additional paths, since LIEMRO is mainly designed for event-driven applications, data reception rate is used to determine whether a newly established path must remain as an active path or not.

Figure 2 (b) demonstrates how LIEMRO behaves when two events occur in the network. As it can be seen, LIEMRO still surpasses the lifetime of single path approach; however, this improvement is not as good as the single event case (Figure 2 (a)). We can justify this as the result of higher inter-node interference and more intense contentions in medium access, especially at the nodes that belong to multiple paths.

C. Latency

Using single path routing with high traffic load results in very long waiting time for medium access. In addition, packet collision worsens the problem because more efforts are needed to transmit data packets. According to Figure 3 (a), using LIEMRO to distribute traffic over multiple paths can provide lower latency compared with single path approach. Regarding this figure, the most significant benefit from multipath routing can be observed at packet interval equal to 0.1 second. As the packet interval increases (i.e., lower traffic), less contention can be made for medium access and this results in similar latency of single path and multipath approaches. As we reduce the packet interval, each path of LIEMRO will be used at its saturated level. At this condition, data reception rate reduces and delivered packets suffer from the delay similar to single path approach. (Note that according to Figure 1(a), in high traffic load, data delivery ratio of LIEMRO is higher than single path approach.)
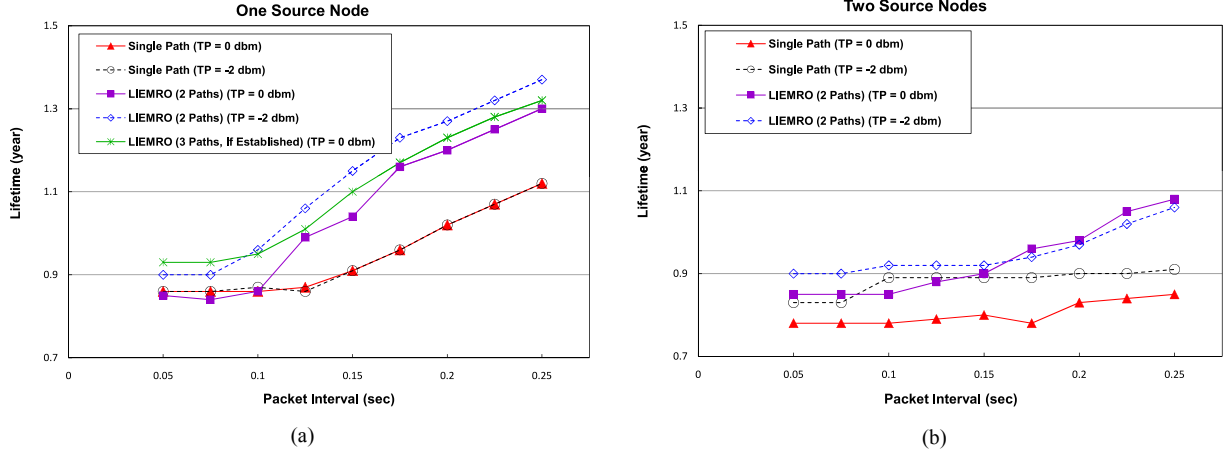
Figure 2. Lifetime of single path routing and LIEMRO versus different traffic loads.
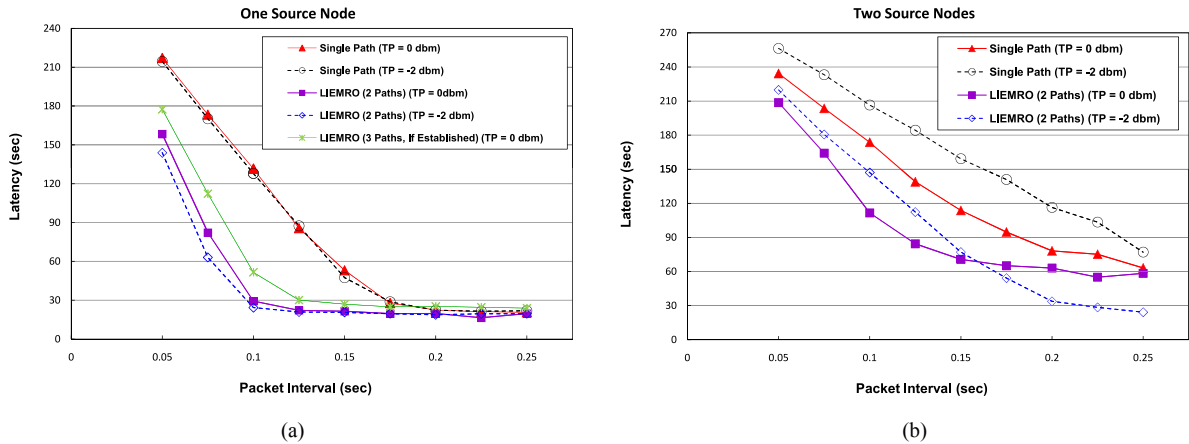

Figure 3. Average end-to-end latency of single path routing and LIEMRO versus different traffic loads.

Figure 3 (a) shows that *LIEMRO (3 Paths)* results in higher latency compared with *LIEMRO (2 Paths)*. However, due to the lower data reception rate of using three paths (Section 0-A), LIEMRO disables the third path and distributes source node's traffic over two paths. Therefore, the described path control mechanism based on data reception rate also results in lower latency.

Figure 3 (b) represents the latency of LIEMRO when two events occur in the network. Generally, more inter-path interference and higher contention for medium access result in upper latency of all approaches, compared with single-event condition. Furthermore, due to the existence of paths with shared nodes, LIEMRO is unable to benefit effectively from data transmission over multiple paths. Yet, LIEMRO provides lower latency than single path approach, even in low traffic load.

In Figure 3 (b), note the meeting point of *LIEMRO (TP=0dBm)* and *LIEMRO (TP=-2dBm)*. In high traffic loads (left hand side of the meeting point), lower latency is achieved by using longer hops. In contrast, in lower traffic

loads (right hand side of the meeting point), using shorter hops resulted in lower latency.

V. CONCLUSION

In this paper, we proposed a multipath routing protocol (LIEMRO) to improve QoS in the terms of data reception rate, lifetime, and latency, for event-driven wireless sensor networks. When an event occurs, LIEMRO tries to establish multiple node-disjoint interference-minimized paths from source to the sink. After creation of each path, it checks if it results in lower data reception rate at the sink node. If so, it disables the recently created path, stops path establishment process, and sends source node's traffic over the previously created paths. Moreover, a load balancing algorithm is used to distribute traffic over each path based on its relative quality to other paths. Simulation results show that LIEMRO can improve data reception rate, lifetime, and latency, compared with single path approach that uses ETX as its routing metric.

According to the simulation results, when multiple events occur in the network, some nodes may be shared by different paths. This issue reduces the efficacy of multipath approach significantly. Therefore, devising a proper n-to-m routing algorithm from n sources to m sinks can improve multipath effectiveness in multiple-event environments.

Applying a joint MAC and physical layer power control mechanism in conjunction with multipath routing protocol can result to further improvement of protocol performance, especially in multiple-event scenarios. However, in this situation, establishing interference-minimized paths needs some assumptions regarding the maximum transmission power in a specific duration of time.

REFERENCES

[1]   J. Tsai and T. Moors, "A Review of Multipath Routing Protocols: From Wireless Ad Hoc to Mesh Networks," *in Proceeding of ACoRN Workshop on Wireless Multihop Networking*, Jul. 2006.

[2]   W. Lou, W. Liu, and Y. Zhang, "Performance Optimization using Multipath Routing in Mobile Ad Hoc and Wireless Sensor Networks," *in Combinatorial Optimization in Communication Networks*, 2005.

[3]   S. Mueller, R. Tsang, and D. Ghosal, "Multipath Routing in Mobile Ad Hoc Networks: Issues and Challenges," *In Performance Tools and Applications to Networked Systems, Volume 2965 of lncs*, 2004, pp. 209-234.

[4]   Y. Wang, H. Lin, and S. Chang, "Interference on Multipath QoS Routing for Ad Hoc Wireless Network," *in ICDCSW'04*, IEEE Computer Society, 2004, pp. 104-109.

[5]   M. Pearlman, Z. Haas, P. Sholander, and S. Tabrizi, "On the Impact of Alternate Path Routing for Load Balancing in Mobile Ad Hoc Networks," *in MobiHOC'00*, Boston, Massachusetts: 2000, pp. 3-10.

[6]   D. Ganesan, R. Govindan, S. Shenker, and D. Estrin, "Highly-Resilient, Energy-Efficient Multipath Routing in Wireless Sensor Networks," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 5, 2001, pp. 11-25.

[7]   C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks," *in MobiCom'00*, Boston, USA: 2000, pp. 56-67.

[8]   A. Nasipuri and S. Das, "On-Demand Multipath Routing for Mobile Ad Hoc Networks," *in ICCCN'99*, Boston, MA: 1999, pp. 64-70.

[9]   M. Marina and S. Das, "On-Demand Multipath Distance Vector Routing in Ad Hoc Networks," *Ninth International Conference on Network Protocols*, 2001, pp. 14-23.

[10]   R. Leung, J. Liu, E. Poon, A. Chan, and B. Li, "MP-DSR: a QoS-aware Multi-path Dynamic Source Routing Protocol for Wireless Ad-Hoc Networks," *in LCN'01*, Tampa, Florida: 2001, pp. 141-132.

[11]   S. Dulman, T. Nieberg, J. Wu, and P. Havinga, "Trade-Off Between Traffic Overhead and Reliability in Multipath Routing for Wireless Sensor Networks," *WCNC Workshop*, New Orleans, Louisiana, USA: 2003.

[12]   B. Deb, S. Bhatnagar, and B. Nath, "ReInForM: Reliable Information Forwarding Using Multiple Paths in Sensor Networks," *in LCN'03*, Los Alamitos, CA, USA: IEEE Computer Society, 2003, pp. 406-415.

[13]   Y.M. Lu and V.W.S. Wong, "An Energy-Efficient Multipath Routing Protocol for Wireless Sensor Networks," *Wiley International Journal of Communication Systems*, vol. 20, 2007, pp. 747-766.

[14]   R Vidhyapriya and Dr P T Vanathi, "Energy Efficient Adaptive Multipath Routing for Wireless Sensor Networks," *IAENG International Journal of Computer Science*, Aug. 2007.

[15]   D.S.J.D. Couto, D. Aguayo, J. Bicket, and R. Morris, "A High-Throughput Path Metric for Multi-Hop Wireless Routing," *in MobiCom'03*, San Diego, CA, USA: ACM, 2003, pp. 134-146.

[16]   W. Ye, J. Heidemann, and D. Estrin, "An Energy-Efficient MAC Protocol for Wireless Sensor Networks," *IEEE Infocom*, New York, NY, USA: 2002, pp. 1567-1576.

[17]   M.Z. Zamalloa and B. Krishnamachari, "An Analysis of Unreliability and Asymmetry in Low-Power Wireless Links," *ACM Trans. Sen. Netw.*, vol. 3, 2007, p. 7.