Regular Article

A Distributed Heuristic Algorithm for Delay Constrained Energy Efficient Routing in Wireless Sensor Networks

Trong-Thua Huynh[1], **Cong-Hung Tran**[2], **Anh-Vu Dinh-Duc**[3]

[1] Ho Chi Minh City University of Technology, Vietnam
[2] Posts and Telecommunications Institute of Technology, Ho Chi Minh city, Vietnam
[3] University of Information Technology, Ho Chi Minh city, Vietnam

Correspondence: Trong-Thua Huynh, htthua@ptithcm.edu.vn
Communication: received 30 May 2016, revised 28 September 2016, accepted 13 November 2016
Online publication: 31 January 2017, Digital Object Identifier: 10.21553/rev-jec.114
The associate editor coordinating the review of this article and recommending it for publication was Dr. Truong Trung Kien.

Abstract– **Besides energy restriction, wireless sensor networks (WSNs) should be able to provide bounded end-to-end delay when they are used to support real-time applications such as early forest fire alarm systems. In this paper, we investigate the problem of finding the least energy consumption route subject to a delay constraint with low computational complexity in such networks. Based on the distance-vector routing approach, which has less computational complexity and message overhead, we propose a distributed heuristic algorithm called Delay Constrained Energy Efficient Routing (DCEER) in order to minimize the total energy consumption while meeting the end-to-end delay requirement. DCEER only requires a moderate amount of information at each sensor node and does not suffer from the excessive running time. We prove that our proposed algorithm always finishes within a finite time and the computation complexity is only $O(n)$, where n is a divisor of the number of sensor nodes. By mathematical proof and simulation, we verify that DCEER is suitable for large-scale WSNs because the number of messages exchanged between sensor nodes are represented by a polynomial function. Furthermore, we evaluate our proposal to compare its performance with related protocols.**

Keywords– **Wireless sensor network, cluster, routing, energy consumption, delay.**

1 Introduction

Energy efficiency is the main objective in the design of the WSN. However, in real-time applications such as early forest fire alarm systems, data should be transmitted from sources to the sink within a limited time. If it exceeds this time, data will not be useful any more. Thus, a trade-off exists between energy consumption and end-to-end delay for such applications. Although many heuristic solutions have been proposed to balance the network delay and the energy consumption in WSNs [1–6], none of them achieves the optimal trade-off.

In-network data aggregation is the technique that processes redundant data and fuses disparate data into a unified data [7]. Clustering algorithms are used in in-network data aggregation to create small data packets that carry all necessary information from sensors monitoring a specific region before forwarding them to the destination [8]. In addition, multi-hop routing protocols can forward data to the destination in a variety of multi-hop routes with high reliability and good balance of energy [9]. Our approach is based on associating clustering [8] and multi-hop routing [9] in order to (i) aggregate in-network data before transmitting them to the sink for reducing the total energy consumption and communication overhead across the network; and (ii) pursue multi-hop data relaying to minimize transmission energy while meeting the end-to-end delay requirement.

There are several techniques to minimize the energy consumption in WSNs based on the clustering approach [8]. However, in this work, we apply the clustering scheme proposed in [6], which uses the Trade-off for Energy and Delay (TED) index to make a trade-off between energy consumption and end-to-end delay. The major contribution of this paper is the proposal of a new multi-hop routing algorithm that can find the least energy consumption route starting from a given cluster-head to the sink within a bounded end-to-end delay.

In [1] and [5], the authors focused on constructing a network topology that aims to make a good trade-off between energy consumption and end-to-end delay so that routing is just forwarding data from any sources to a given destination with a fixed topology. In this paper, we instead investigate into route selection based on network conditions at different time. Besides, instead of routing based on the aggregate cost function of both delay and energy consumption metrics as in [6], we focus on route selection that is based on each metric separately but mutually bounded.

Many solutions [10–18] have been proposed to tackle the problem of delay constrained energy efficient routing in WSNs with various degrees of success. This problem is to minimize the total energy consumption

E_{total} of a route R_i, while keeping the end-to-end delay D_{ete} under a given constraint Δ. Specifically, we want to

$$\min E_{\text{total}}(R_i) : D_{\text{ete}}(R_i) \leq \Delta, \forall R_i \in \Re(\text{CH}_i, \text{SINK}), \quad (1)$$

where $\Re(\text{CH}_i, \text{SINK})$ is the set of routes from the cluster-head i to the destination node, SINK. In our model, there is only one destination node. This node receives all data from the sensor nodes in the whole network.

Unfortunately, because this problem has been proven to be NP-hard [19], there exist no algorithms that can find the optimal solution and run in polynomial time. The above-mentioned solutions did not try to solve this complex problem, instead they defined simpler problems. In this paper, we focus on a simplified problem by considering the class of applications having only a single destination. Our algorithm uses combined routes instead of combined metrics in order to reduce the search space.

The novelty in our proposal is mainly in the route discovery algorithm. It only uses a moderate amount of information from neighboring nodes instead of the whole network, and the routing is effective, fast convergent and with less overhead. To achieve so, we define several special messages for exchanging information among neighboring nodes as well as a simple entry to store the exchanged information. These assist the routing in quickly determining the best next node of the optimal route with low computational complexity.

The remainder of the paper is organized as follows. Section 2 discusses existing proposals for this problem. Section 3 briefly describes the pseudo code of the clustering algorithm proposed in [6]. Section 4 presents the multi-hop routing algorithm for DCEER and analyzes its convergence and complexity. Section 5 shows the simulation results for DCEER's performance. Section 6 concludes the paper.

2 Related Works

Huynh *et al.* proposed a multi-hop routing scheme to balance energy consumption and network delay in [1]. An energy* delay routing algorithm is applied to sensor nodes within each three-hop cluster while an energy-efficient chain construction algorithm is applied to cluster-heads to construct energy-efficient chains from the cluster-heads to the base station. In [5], the authors proposed another energy efficient delay-aware routing algorithm for a multi-layer WSN, wherein cluster-heads at each layer are interconnected as de Bruijn graph to improve network delay, energy consumption, and system reliability. Recently, in [6], the authors have proposed a new distributed clustering method to determine the best cluster-head for each cluster by controlling adjustment parameters. In addition, they have proposed a new aggregated cost function used in the multi-hop routing algorithm from cluster-heads to sink for a trade-off between energy consumption and end-to-end delay.

In HEED [10], the cluster-heads are selected periodically based both on the residual energy of a sensor node and the distance from each sensor node to its neighboring nodes. HEED can achieve a uniform cluster-head distribution across the whole network, but it needs too many iterations, incuring a high overhead. Based on HEED's architecture, Akkaya and Younis proposed a routing protocol [11] that finds an energy-efficient route along which end-to-end delay requirements are met. This protocol divides data packets into two categories based on the delay and bandwidth requirements, puts them into the corresponding queues, and forwards to the sink. Their approach, however, did not take into consideration the delay factors that can occur due to channel contention at the MAC layer.

In [12], authors proposed an energy efficient and delay constrained routing protocol based on the length of data packet and queue length to reduce network congestion. This protocol allows time-sensitive packets to select the shortest path for reducing the end-to-end delay while other data packets to select the neighbor nodes for forwarding the data to the sink for an energy balance. In [13], Han *et al.* proposed a minimum spanning tree with the Wiener index for WSNs, in which base nodes are mobile. For small-scale WSNs, the branch and bound algorithm used to minimize the search space. For large-scale WSNs, the simulated annealing algorithm was used to ignore the local optimal solutions and toward better optimal solution. This proposal outperforms the minimum spanning tree in terms of energy consumption and network delay. With the same objective, Niu *et al.* studied minimum delay and energy efficient flooding tree construction for WSNs whose links are unreliable [14]. The problem was formulated as a minimum spanning tree problem with an undetermined delay constraint. A distributed heuristic algorithm based on the known delay constraint was proposed to solve this problem. Although their proposed algorithm achieves a good balance between flooding delay and energy consumption, it requires a lot of overhead to construct the minimum spanning tree.

Nadeem *et al.* proposed a gateway-based energy-aware multi-hop routing protocol for WSNs called M-GEAR in [15]. To save energy, the proposed protocol divides the sensor nodes in different regions, each region uses a different routing architecture. In addition, this protocol deploys a gateway located in the sensing region to support the sensor nodes in clustering and routing. Based on the distance from each sensor node to the gateway and to the base station, the protocol decides to carry out either direct transmission or indirect multi-hop transmission. However, this protocol can only be applied to stationary sensor networks; it is difficult to implement such network topology. Moreover, the handling of clustering and setting up a TDMA schedule is dependent on a gateway; it cannot be deployed in distributed networks. Similarly, Guidoni *et al.* proposed a routing protocol based on topologies for heterogeneous WSNs called RouT in [16]. With the same physical network architecture, the proposed

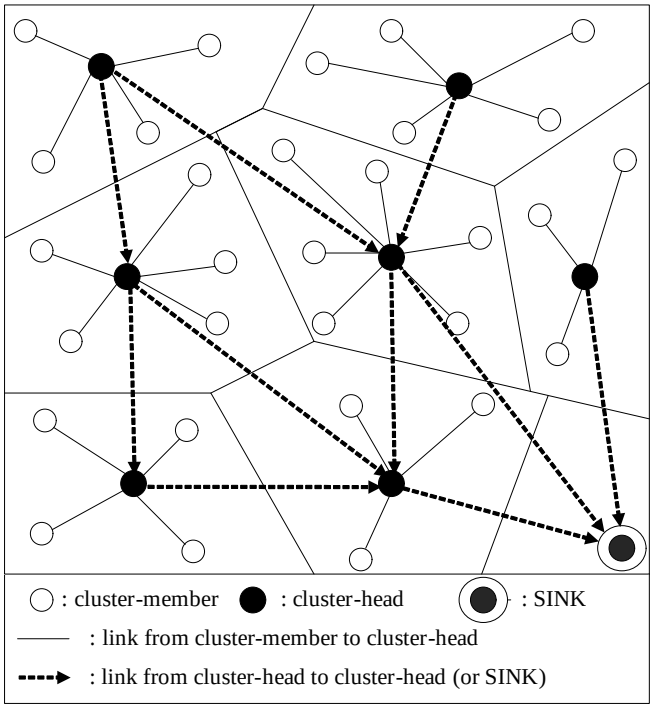

Figure 1. Hierarchical wireless sensor network model.

protocol creates different logical topologies depending on the capacities of sensor nodes. They are divided into two types as L-sensors (low capacity) and H-sensors (high capacity). Each logical topology is set based on the delay of links connected between the H-sensors. In this way, each logical topology creates a balance between energy consumption and network delay. However, the messages exchange between sensor nodes to establish the logical topologies amount to more overhead and end-to-end delay.

DEAR (delay bounded adaptive energy constrained routing) proposed in [17] is a multi-path routing protocol. It considers many parameters such as reliability, delay and energy consumption. This protocol allows packets to be continuously distributed across the network even if the paths are going to crash. It balances the delay among different paths by using a polynomial-time algorithm to solve a multi-objective optimization problem. In [18], Yao *et al.* designed an energy efficient delay aware protocol to balance network lifetime, called EDAL. They proposed a centralized heuristic to reduce computational complexity and a distributed heuristic to make the algorithm suitable for large-scale WSNs. This algorithm was also extended, for reducing the total energy consumption for applications with loose delay constraints.

3 DCEER: Distributed Clustering Scheme

Consider a set of sensor nodes dispersed in a field as illustrated in Figure 1. We employ the hierarchical network model in which sensor nodes are distributed in clusters. Each cluster itself elects a cluster-head that aggregates data from its cluster-member nodes and sends the aggregated data to SINK in multi hops.

Algorithm 1 Pseudo codes of clustering algorithm for each sensor node

Input: ADV message from SINK
Output: *NodeRole*

1: Calculate the approximate distance from sensor node to SINK, d_{toSINK}
2: Wait for $\tau = \frac{1}{E}$
3: broadcast ADV message to neighbors
4: **if** Received ADV message from other neighbors **then**
5: **for** $j = 0$ to N_i **do**
6: **if** $E_i \leq E_j$ **then**
7: flag $= 1$
8: break
9: **end if**
10: **end for**
11: **if** flag $== 1$ **then**
12: *NodeRole* $=$ Cluster-member
13: **end if**
14: **end if**
15: Wait for $\omega = \dfrac{1}{\text{TED}_i}$
16: **if** Received JCR message **then**
17: Send ACK message to its cluster-head
18: *NodeRole* $=$ Cluster-member
19: **else**
20: Broadcast JCR message to others
21: *NodeRole* $=$ Cluster-head
22: **end if**
23: Broadcast NCR message to other cluster-heads
24: Update nodal residual energy
25: Return *NodeRole*

DCEER operates in consecutive rounds. Each round is separated into two phases: (i) network organization, which establishes the cluster network topology, and (ii) data transmission, which finds the best route to transmit data from cluster-heads to SINK. In the first phase, DCEER uses the distributed clustering scheme proposed in [6] to set up clusters.

The pseudo codes of this algorithm is described in Algorithm 1. In line 3, the ADV message is used to calculate the distance from each sensor node to other sensor nodes. In line 6, E_i and E_j are the residual energy of sensor nodes i and j, respectively. In line 15, the TED index is determined by

$$\text{TED}_i = \left(\frac{E_i}{E_{\text{total}}}\right)^{\alpha} + \left(\frac{1}{d_{i,\text{SINK}}}\right)^{\beta}, \tag{2}$$

where E_i is the residual energy of the cluster-head candidate i, E_{total} is the cumulative energy of the other cluster-head candidates it has received from ADV messages, $d_{i,\text{SINK}}$ is the distance from cluster-head candidate i to SINK. Values of α and β are in the range of $[0, 1]$ and $\alpha + \beta \neq 0$. In lines 16 and 20, the JCR (Join Cluster Request) message is used to determine whether a sensor node has joined the cluster. In line 23, the NCR (Neighbor Cluster-head Request) message is used to determine the neighbor cluster-heads.

4 DCEER: Multi-hop Routing

After the clusters have been set up, each cluster-member node turns off the radio until its allocated transmission time for sending data to the cluster-head. The cluster-head keeps its receiver on to receive data from its cluster-member nodes. After all data have been received, the cluster-head aggregates all data packets into a single packet to reduce redundancy and transmission energy. It then discovers the best route to transfer the aggregated data to SINK. This route must have the least total energy consumption while keeping the end-to-end delay under a given value Δ.

4.1 Route Discovery Algorithm

Each cluster-head i must have the following information to find the best route to delivery data to the SINK.

(i) $\text{nextCH}_{\text{le}}$: the next cluster-head (node j) that makes cluster-head i consume the least energy in sending data to node j, that is,

$$\text{nextCH}_{\text{le}} = \min_{j \in N_i} E_{\text{Tx}}(i,j), \tag{3}$$

where N_i are the neighbors of cluster-head i, $E_{\text{Tx}}(i,j)$ is energy consumed in transmitting an l-bit data packet from cluster-head i to cluster-head j over a distance of d_{ij}, and is defined as

$$E_{\text{Tx}}(i,j) = \begin{cases} lE_{\text{elec}} + l\epsilon_{\text{fs}} d_{ij}^2, & \text{if } d_{ij} < d_0, \\ lE_{\text{elec}} + l\epsilon_{\text{mp}} d_{ij}^4, & \text{if } d_{ij} \geq d_0, \end{cases} \tag{4}$$

where E_{elec} is an electronic energy consumption factor, ϵ_{fs} and ϵ_{mp} are the amplifier energies to maintain an acceptable signal-to-noise ratio, $d_0 = (\epsilon_{\text{fs}}/\epsilon_{\text{mp}})^{1/2}$ is the reference distance between the transmitter and the receiver.

(ii) $\text{nextCH}_{\text{ld}}$: the next cluster-head (node j) that makes cluster-head i take the least delay in sending data to node j, that is,

$$\text{nextCH}_{\text{ld}} = \min_{j \in N_i} D(i,j), \tag{5}$$

where N_i are the neighbors of cluster-head i, $D(i,j)$ is a measure of the delay a packet experiences when traversing the link from cluster-head i to cluster-head j, which is defined as

$$D(i,j) = D_Q + D_T + D_P, \tag{6}$$

where D_Q, D_T and D_P are, respectively, the queuing delay per cluster-head node, the transmission delay, and the propagation delay. According to [6], we have:

$$D(i,j) = \frac{1}{\mu - \lambda} + \frac{l}{\psi} + \frac{d_{ij}}{\gamma}, \tag{7}$$

where μ is the service rate which is an exponential stochastic variable, λ is the rate of entry for new packets which is a Poisson stochastic variable, l is the packet size (in bits), ψ is the link bandwidth (in bps), d_{ij} is the length of the physical link from cluster-head i to cluster-head j, and γ is the propagation speed in medium (in m/s).

(iii) $\text{ld}(i,\text{SINK})$: the least time required for sending data from cluster-head i to the SINK. This value is calculated similarly to Equation (7) except that d_{ij} is replaced with $d_{i,\text{SINK}}$, which is the distance from cluster-head i to the SINK. This distance is calculated by using the RSSI (Received Signal Strength Indicator) value it received from the first step of the cluster set-up phase.

At any time, if cluster-head i has data to send to the SINK, it initiates route discovery by checking the $\text{ld}(i,\text{SINK})$. If this value is greater than the bounded delay, there does not exist any route between cluster-head i and the SINK that meets the delay constraint. It stops the algorithm execution and does not send data out. If, however, $\text{ld}(i,\text{SINK}) \leq \Delta$, i.e., the routes meet the delay constraint between cluster-head i and the SINK, it continues execution until a route is found. In this case, cluster-head i determines the next cluster-head, j, that satisfies Equation (3). Then, cluster-head i sends the Least Delay Request (LDR) message to its neighboring cluster-head j in order to get the value of $\text{ld}(j,\text{SINK})$. Cluster-head j returns the newest value of $\text{ld}(j,\text{SINK})$ to cluster-head i. Cluster-head i checks the following inequality:

$$D(\text{initCH}, \text{currCH}) + D(i,j) + \text{ld}(j,\text{SINK}) \leq \Delta, \tag{8}$$

where $D(\text{initCH}, \text{currCH})$ is the time elapsed from the initiating cluster-head of this route to the current cluster-head (in this case, it is the cluster-head i).

If this inequality is satisfied then there exist valid delay constrained routes from cluster-head i to the SINK that meet the delay constraint Δ and they use the link (i,j) to follow the least energy route. Cluster-head i selects cluster-head j as the next node of route toward the SINK. If the inequality is not satisfied, cluster-head i selects another cluster-head, k, as the next node by using Equation (5) to follow the least delay route. This selection ensures that the route from the current cluster-head, i, to the SINK is a part of at least one route from the initiating cluster-head to the SINK that meets the delay constraint, otherwise, the current cluster-head could not have been selected in a previous step.

After selecting the link (next node) to follow, the current cluster-head, i, creates a routing table entry for this route. It contains the information as shown in Figure 2.

For the current cluster-head (CH_i) of this route identified by *routeCODE*, initCH is the initiating cluster-head of the route, prevCH is the neighboring cluster-head that precedes CH_i on the route, and nextCH is the neighboring cluster-head that follows CH_i on the route. nextCH is $\text{nextCH}_{\text{le}}$ if (8) is satisfied, and $\text{nextCH}_{\text{ld}}$, otherwise. $D(\text{initCH}, \text{currCH})$ is time elapsed from initCH to the current cluster-head of this route.

routeCODE	initCH	prevCH	nextCH	D(initCH,currCH)

Figure 2. Routing table entry.

routeCODE	initCH	△	D(initCH,nextCH)	D(initCH,currCH)

Figure 3. Constructed Cluster-heads Route (CCR) message.

Then, the current cluster-head creates a new delay value, $D(\text{initCH}, \text{nextCH})$, which is the delay along the constructed route from the initiating cluster-head to the next cluster-head. This value is calculated by adding $D(\text{currCH}, \text{nextCH})$ to the current delay $D(\text{initCH}, \text{currCH})$, that is,

$$D(\text{initCH}, \text{nextCH}) = D(\text{initCH}, \text{currCH}) + D(\text{currCH}, \text{nextCH}). \quad (9)$$

After all, it sends a Constructed Cluster-heads Route (CCR) message to the next cluster-head. This message contains the information as shown in Figure 3.

In Figure 3, Δ is the bounded delay of the route identified by *routeCODE*. If any cluster-head j, which is not the SINK, receives a CCR message, it becomes the current cluster-head of this route. This cluster-head j selects the next cluster-head, creates a routing table entry and constructs the route in a similar manner as described above. At this time, the prevCH value in the routing table entry of cluster-head j is the cluster-head i that sent it a CCR message. When the SINK receives a CCR message, it sends an acknowledgement (ACK) message back to the initiating cluster-head of that route. When the initiating cluster-head receives the ACK message, it starts to transmit data to the SINK following the route which has been previously discovered. The selection of the next cluster-head is detailed in Algorithm 2, implemented at each cluster-head.

Algorithm 2 Find next cluster-head that consumes least energy and meets end-to-end delay constraint - *Find_NextCH(Input,Output)*

Input: prevCH, initCH, SINK, Δ, $D(\text{initCH}, \text{currCH})$
Output: nextCH

The current cluster-head is the initiator
1: **if** (prevCH = null | $D(\text{initCH}, \text{currCH}) = 0$ | initCH $= i$) **then**
2: **if** $\text{ld}(i, \text{SINK}) > \Delta$ **then**
3: nextCH = null
4: exit(1)
5: **end if**
6: **end if**
Every cluster-head will execute to construct the route
7: **for** $j = 0$ to N_i **do**
8: $\text{nextCH}_{\text{le}} = \min E_{\text{Tx}}(i, j)$
9: **end for**
10: temp = NodeID($\text{nextCH}_{\text{le}}$)
11: **if** $(D(\text{initCH}, \text{currCH}) + D(i, \text{temp}) + \text{ld}(\text{temp}, \text{SINK})) \leq \Delta$ **then**
12: nextCH = temp
13: **else**
14: **for** $j = 0$ to N_i **do**
15: $\text{nextCH}_{\text{ld}} = \min D(i, j)$
16: **end for**
17: $\text{nextCH} = \text{NodeID}(\text{nextCH}_{\text{ld}})$
18: **end if**
19: $D(\text{initCH}, \text{nextCH}) = D(\text{initCH}, \text{currCH}) + D(i, \text{nextCH})$
20: **return** nextCH

4.2 Convergence and Complexity of DCEER

We verify the convergence of DCEER by proving that it can always finishes within a finite time. In addition, we prove that the computation complexity of DCEER is a constant function as well as its message overhead is a polynomial function.

We denote the route created by k cluster-heads as R_k, that is $R_k : \text{CH}_0 \rightarrow \cdots \rightarrow \text{CH}_k$. Route R_{k+1} is created by adding a link to R_k. Let $\Im_k = \{\text{CH}_0, \ldots \text{CH}_k\}$ be the set of cluster-heads of route R_k, and $\Re_k = \{R_0, \ldots, R_k\}$ the set of routes constructed at the k^{th} step.

Theorem 1. *Algorithm 2 always finds either the route that consumes the least energy and meets the end-to-end delay constraint if such a route exists or no route within a finite time.*

Proof: If there exists no route from a given cluster-head i to the SINK, i.e., $\text{ld}(i, \text{SINK}) > \Delta$ (line 2), the algorithm terminates immediately after line 4 without constructing the route. Otherwise, i.e., $\text{ld}(i, \text{SINK}) \leq \Delta$, the algorithm continues execution until a route is found. It is proven by induction on the number of the constructed routes as follows.

For $k = 0$, cluster-head i is the initiator of the route and it is connected directly to the SINK, i.e., $R_0 = \text{CH}_0$, $\Re_0 = R_0$. Since $D(R_0) = 0$ and $\text{ld}(\text{CH}_0, \text{SINK}) \geq \Delta$, then algorithm will definitely stop with nextCH = SINK.

Suppose now that

$$D(R_k) + \text{ld}(\text{CH}_k, \text{SINK}) \leq \Delta. \quad (10)$$

We have the following two cases.

Case 1: R_{k+1} extends a route $R_k \in \Re_k$ by adding the link that has chose $\text{nextCH}_{\text{le}}$ as the next cluster-head from CH_k to follow the least energy route. The following inequalities must be satisfied:

$$D(R_k) + D(\text{CH}_k, \text{CH}_{k+1}) + \text{ld}(\text{CH}_{k+1}, \text{SINK}) \leq \Delta \quad (11a)$$

$$D(R_{k+1}) + \text{ld}(\text{CH}_{k+1}, \text{SINK}) \leq \Delta. \quad (11b)$$

Case 2: R_{k+1} extends a route $R_j \in \Re_k$ $(0 \leq j \leq k)$ by adding the link that has chose $\text{nextCH}_{\text{ld}}$ as the next cluster-head from CH_j to follow the least delay route. In this case, we have

$$\text{ld}(\text{CH}_j, \text{SINK}) = D(\text{CH}_j, \text{CH}_{k+1}) + \text{ld}(\text{CH}_{k+1}, \text{SINK}).$$

Thus, the following inequalities must be satisfied:

$$D(R_j) + D(\text{CH}_j, \text{CH}_{k+1}) + \text{ld}(\text{CH}_{k+1}, \text{SINK}) \leq \Delta \quad (12a)$$

$$D(R_{k+1}) + \text{ld}(\text{CH}_{k+1}, \text{SINK}) \leq \Delta. \quad (12b)$$

In both cases, it is clearly that the delay of route R_{k+1} is upper bounded by Δ since inequalities (11) and (12) are the same. ■

Theorem 2. *The route constructed by Algorithm 2 from a given cluster-head i to the SINK is loop-free.*

Proof: Let $\Im_j$ be the set of cluster-heads of route R_j. A loop can only be created if there exists a link that is added by connecting cluster-head j with another cluster-head, k, where $\mathrm{CH}_k \neq \mathrm{CH}_j$ and $\mathrm{CH}_k \in \Im_j$. However, a routing table entry of cluster-head k indicates that there exists a link between j and k for this route. This new entry will not be added in the routing table. Thus, the route constructed by DCEER is loop-free. ■

Theorem 3. *The computation complexity of DCEER is a linear function of n, i.e. $O(n)$, where n is a divisor of the number of sensor nodes in a WSN.*

Proof: For Algorithm 1, it is easy to see that each node will only execute a fixed number of simple calculations (calculate the approximate distance d_{toSINK} in line 1, calculate the first timer τ in line 2, calculate the second timer ω in line 15, and update nodal residual energy in line 24) as well a fixed number of operations (broadcast the ADV message to neighbors in line 3, send the ACK message back to its cluster-head in line 17, and broadcast the JCR and NCR messages to neighbors in lines 20 and 23). Besides, it only performs checking a fixed number of simple conditions (lines 4, 6, 11, and 16). The FOR loop is executed repeatedly in $N_i = N/K$, where N is the number of sensor nodes, K is the expected number of clusters, and N_i is the average number of sensor nodes within each cluster for a WSN. Therefore, the computation complexity of Algorithm 1 is a linear function $O(m)$, where m is the average number of sensor nodes within each cluster of a WSN.

In addition, the computation complexity of Algorithm 2 at any cluster-head for constructing a route between a given cluster-head i and the SINK is $O(N_i)$, where N_i is the number of neighboring cluster-heads of node i. This is because each time a cluster-head receives a CCR message, it just performs checking the delay condition (lines 1, 2, and 11) and calculates the minimum values of delay and energy for each outgoing link (lines 8, 15, and 19) in order to select the next cluster-head within a limited number of neighbors, N_i. In the worst case, each calculation in lines 8 and 15 takes N_i times and thus has complexity of $O(N_i)$.

Consequently, the computation complexity of DCEER is a linear function $O(m) + O(N_i) \approx O(n)$, where n is a divisor of the number of sensor nodes in a WSN. ■

Theorem 4. *The message overhead of DCEER is a polynomial function $O(N^2)$, where N is the number of sensor nodes.*

Proof: For Algorithm 1, the number of messages needed to establish the clusters depends upon the ADV, JCR and ACK messages (lines 3, 17, and 20). For a network of N sensor nodes, K expected clusters, consider the nodal operation in line 3, the maximum number of ADV messages which are broadcast to neighbors is $K(N_i(N_i+1)/2) \approx KN_i^2$, where N_i is the average number of sensor nodes within each cluster. Because $N_i = N/K$, the maximum number of ADV messages is approximated to N^2/K.

Considering the operation in line 17, the maximum number of ACK messages which are sent to the cluster-heads is $(N-K)$. Similarly, for line 20, the maximum number of JCR messages which are broadcast to neighbors is KN_i. Because $N_i = N/K$, the maximum number of JCR messages is approximated to N. Therefore, in the worst case, the total number of messages exchanged in the process of setting up clusters will be $(N^2/K) + N$. In other words, the message overhead of Algorithm 1 is $O(N^2)$.

Furthermore, in Algorithm 2, it is easy to see that the number of messages needed to construct a route is proportional to the number of links in that route. For a network of K cluster-heads, the longest loop-free route contains at most K cluster-heads and $(K-1)$ links. Thus, constructing this loop-free route requires $O(K)$ messages in the worst case.

Consequently, the message overhead of DCEER is a polynomial function $O(N^2) + O(K) \simeq O(N^2)$. ■

5 Simulation Results

The maximum one-hop delay depends on the least delay from the farthest node to the sink node. To guarantee that cluster-head nodes can find the sufficient routes to the sink node, the bounded end-to-end delay should not be less than this least delay. In addition, to reduce interference among neighboring sensor nodes and to ensure that the cluster-head nodes can find the sufficient routes to the sink without fail, we set the maximum transmission range value based on the RSSI value. Based on [20], we rewrite the following formula to express the relationship between the distance (transmission range) and RSSI in a WSN.

$$d = 10^{(E_{\mathrm{Tx}} - \mathrm{RSSI})/(10n)}, \tag{13}$$

with $n = 2$ for the free space. Then, the maximum transmission range (distance) depends on the RSSI value and E_{Tx} defined in Equation (4). In our simulation scenarios, we set the minimum RSSI value to -91 dBm.

Our simulation is implemented for cluster-based WSNs using Castalia simulator [21] on a Ubuntu platform. We set the parameters common to all experiments as follows. The data packet size is 30 bytes, $\lambda = 3$, $\mu = 6$, initial energy of node is 1 joule, $E_{\mathrm{elec}} = 50$ nJ/bit, $\epsilon_{\mathrm{fs}} = 10$ pJ/bit/m^2, $\epsilon_{\mathrm{mp}} = 0.0013$ pJ/bit/m^4, $E_{\mathrm{aggr}} = 5$ nJ/bit, $\psi = 250$ bps, $\gamma = 3 \times 10^8$ m/s, and $\alpha = \beta = 0.5$. Other parameters will be set in accordance with each particular experiment.

In the first simulation scenario, we measured the average number of messages required to construct energy efficient routes that meet the bounded delay constraint, Δ. This simulation was performed for different network sizes of 100, 200, 300, 400, and 500 sensor nodes. For each network size, we varied the delay constraint as

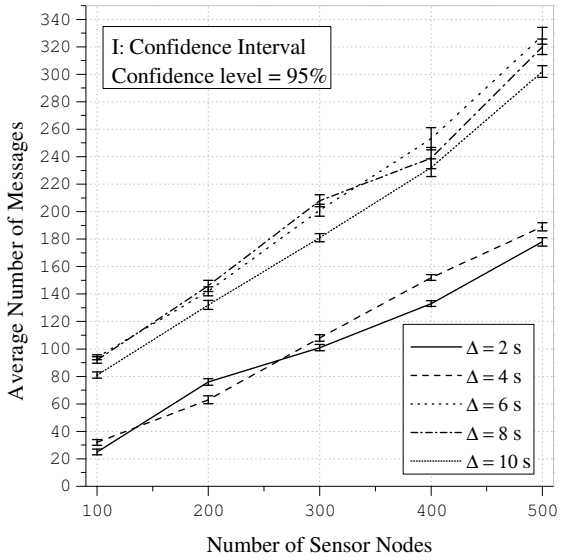

Figure 4. Average number of messages versus network size.

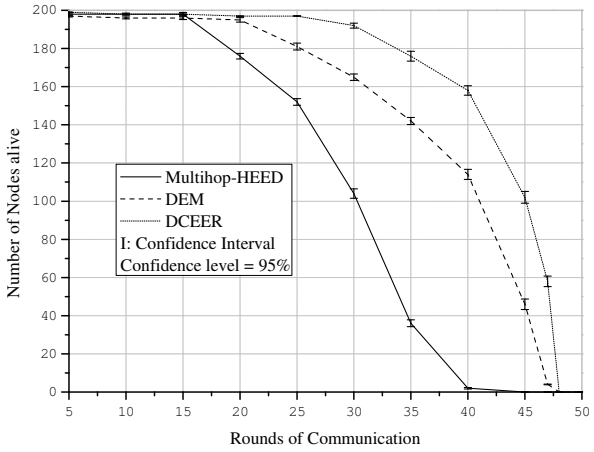

Figure 5. Performance of Multihop-HEED, DEM and DCEER on Number of Nodes alive with respect to given delay constraint.

2, 4, 6, 8, and 10 s. For each delay constraint value, we performed five experiments with the same network size. Applying the DCEER algorithm, we measured the average number of messages exchanged between the sensor nodes. For a confidence level set to 95%, confidence intervals were indicated for each delay constraint value correspondingly.

Figure 4 shows the average number of messages versus the size of the network for five different values of the bounded delay. All five curves indicate that the average number of messages grew steadily with the network size. For any of the bounded delay values, extending the size of the network lead to an increase in the average number of messages by a maximum of 10%. This makes DCEER suitable to deploy for large-scale WSNs.

In addition, it is obvious that the smaller the bounded end-to-end delay the fewer the number of links per route was, although the number of messages exchanged between the cluster-heads to construct the route can not be fewer correspondingly. They were even more in some cases. However, on average, the number of messages exchanged between the cluster-heads decreased as the bounded end-to-end delay decreased. That is because the cluster-head is forced to follow the least delay route direction most of the time when the bounded delay was small. For a network size of 400 sensor nodes, with a bounded delay $\Delta = 2$ s, the number of messages exchanged between the cluster-heads was the smallest, for example, more or less 133 messages in Figure 4. When Δ was increased to 6 s, the number of exchanged messages is the largest, more or less 253 messages. However, when Δ was increased to 10 s, the number of exchanged messages was reduced, more or less 232 messages. The reason is that 6 s is a reasonable value of Δ, and DCEER may be able to follow the least energy route direction at some cluster-heads and to follow the least delay route direction at others. This causes an increase the average number of messages exchanged between the cluster-heads to select the next node. Besides, for a relaxed value of $\Delta = 10$ s, the cluster-head is able to follow the least energy route direction most of the time without breaking the delay constraint. Therefore, this reduces the number of messages exchanged between the cluster-heads.

In the second simulation scenario, we evaluated the performance of DCEER, in comparison with Multihop-HEED [10] and DEM [6], in terms of network lifetime. We implemented the three protocols for the same context with five experiments repeatedly for each simulation scenario. The network consists of 200 sensor nodes distributed in a field with dimensions 100 m $\times$ 100 m. The SINK is located at $(0, 0)$. For a confidence level set to 95%, the results are shown with confidence intervals in Figure 5. To evaluate the network lifetime among protocols, we estimated the number of alive nodes for each round by implementing the experiments up to 50 rounds and fixing Δ at 6 s.

The following analysis is used to explain the results obtained in Figures 5 and 6. In the data transmission phase, the cluster-member nodes only need to send data to the cluster-head. Therefore, the energy consumption of each cluster-member node j is given by

$$E_{\text{mem}}(j) = l(E_{\text{elec}} + \epsilon_{\text{fs}} d^2(j)), \tag{14}$$

where l is the data packet size, E_{elec} is an electronic energy consumption factor, ϵ_{fs} is the amplifier energy to maintain an acceptable signal-to-noise ratio, and $d(j)$ is the distance from cluster-member node j to its cluster-head.

In addition, the cluster-head needs to aggregate all intra-cluster data from its cluster-member nodes and forward the aggregated data to other cluster-heads. Therefore, its energy consumption is given by

$$E_{\text{CH}}(i) = E_R(i) + E_A(i) + E_F(i), \tag{15}$$

where $E_R(i)$ is the energy of cluster-head i consumed to receive all intra-cluster data, $E_A(i)$ is the energy of cluster-head i consumed to aggregate all intra-cluster data, and $E_F(i)$ is the energy of cluster-head i consumed to forward l-bit data to other cluster-heads or the SINK, which is derived from Equation (4). They are defined

by the following:

$$E_R(i) = lE_{\text{elec}}(\text{size}_{\text{CH}}(i) + F), \tag{16}$$

$$E_A(i) = \text{size}_{\text{CH}}(i)E_{\text{aggr}}l, \tag{17}$$

$$E_F(i) = \begin{cases} l(E_{\text{elec}} + \epsilon_{\text{fs}}d_{ij}^2)(1+F), & \text{if } d_{ij} < d_0, \\ l(E_{\text{elec}} + \epsilon_{mp}d_{ij}^4)(1+F), & \text{if } d_{ij} \geq d_0, \end{cases} \tag{18}$$

where $\text{size}_{\text{CH}}(i)$ is the number of cluster-member nodes which belong to the cluster-head i, E_{elec} is a data aggregation factor, F is the number of forwards, d_{ij} is the distance from cluster-head i to its next cluster-head j as defined in Equation (4). Then, the total energy consumption for each round is given by

$$E_{\text{total}} = \sum_{i=1}^{K} E_{\text{CH}}(i) + \sum_{j=1}^{N-K} E_{\text{mem}}(j), \tag{19}$$

where K is the number of cluster-heads, N is the number of sensor nodes in the network.

In Multihop-HEED, each node i elects itself to become a cluster-head with probability $\text{CH}_{\text{prob}}(i) = C_{\text{opt}}(E_i/E_{\text{max}})$, where E_i is the residual energy of node i, E_{max} is the reference maximum energy of node i, and C_{opt} is the optimal number of cluster-heads which is set as an initial percentage of cluster-heads among all N sensor nodes. For Multihop-HEED operation, Dynamic Source Routing [22] is used to communicate among the cluster-heads, or between the cluster-heads and the SINK. However, it selects the most energy efficient routes to forward requests and replies regardless of the end-to-end delay requirement of the application. Moreover, both DCEER and DEM are inter-cluster multi-hop routing protocols that take into account the end-to-end delay in routing decision. Notwithstanding, to reduce the search space and save energy, DCEER uses combined routes instead of combined metrics as proposed in [6], wherein the cost function combines both delay and energy consumption. Therefore, the total energy consumed by the data transmission using DCEER is significantly less than that using Multihop-HEED but slightly less as compared to using DEM. This results in faster death of sensor nodes after each round for both Multihop-HEED and DEM as compared to DCEER as shown in Figure 5. Thus, DCEER prolongs the network lifetime better than both Multihop-HEED and DEM.

In the third simulation scenario, to evaluate the energy efficiency and the data transmission reliability, we implemented five experiments with the same network size but different values of the bounded delay, $\Delta = 2, 4, 6, 8, 10$, respectively. For a confidence level set to 95%, the result is shown with confidence intervals in Figures 6 and 7.

Energy efficiency is the ratio of the total achieved data by the SINK to the total energy consumption. In Figure 6, because Multihop-HEED does not involve the end-to-end delay constraint its energy efficiency was almost unaffected by the bounded end-to-end delay, whereas DCEER clearly showed an impact of the bounded end-to-end delay on the energy efficiency. In particular, when the end-to-end delay is bounded very tightly ($\Delta = 2, 4$), DCEER was less efficient than

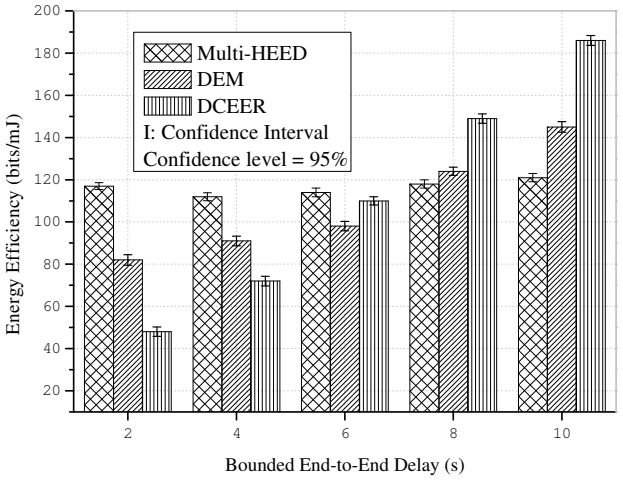

Figure 6. Energy efficiency of Multihop-HEED, DEM and DCEER over end-to-end delay constraints.

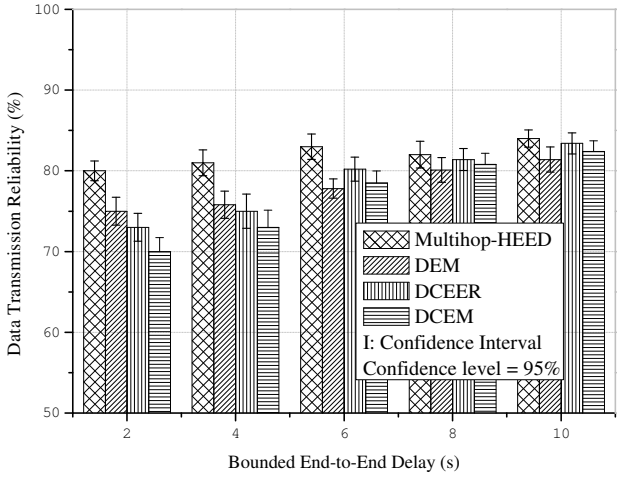

Figure 7. Data transmission reliability of Multihop-HEED, DEM and DCEER over different delay constraints.

DEM and Multihop-HEED in terms of energy consumption. However, when this constraint was loose ($\Delta = 6, 8, 10$), DCEER was more energy-efficient than the other protocols. That is because Multi-HEED not only allows sensors to send data to the SINK whenever they have data regardless of the delay requirement of the application but also elects the cluster-heads upon the nodal residual energy regardless of the distance between sensor nodes. In addition, DEM and DCEER use the same cluster scheme, i.e., both of them elect the cluster-heads upon nodal residual energy and the distance between sensor nodes. However, because DEM associates the delay factor into the cost function, it is not affected by the delay constraint of the application. Therefore, when the end-to-end delay constraint was relaxed, DCEER found more energy efficient routes from any sensor node to the SINK than DEM and Multihop-HEED did.

Data transmission reliability is the percentage of total archived data by the SINK to the total deliverable data. In Figure 7, because Multihop-HEED allows sensors to send data to the SINK whenever they have data regardless of the delay requirement of the application,

the data transmission reliability was quite high (more than 80%). In addition, the data transmission reliability of Multihop-HEED was higher than DEM, DCEM and DCEER, especially, when the end-to-end delay was bounded very tightly ($\Delta = 2, 4$). However, when the end-to-end delay constraint was loose ($\Delta = 6, 8, 10$), DCEM, DCEER and Multi-hop HEED had similar data transmission reliability. Furthermore, since DEM does not consider the end-to-end delay as an independent factor, its objective is just to find the route to balance the energy consumption and the end-to-end delay, and thus its data transmission reliability was much lower than that of DCEM, DCEER and Multihop-HEED.

6 Conclusion

In this paper, we have proposed a new distributed heuristic algorithm called DCEER. It not only elects the best cluster-heads in terms of energy efficiency and network delay but also looks for the least energy consumption routes subject to an end-to-end delay constraint with low computational complexity and message overhead in large-scale WSNs. We have proved that our proposed algorithm always finishes within a finite time as well as its computational complexity and message overhead are a constant function and a polynomial function, respectively. In addition, we have also verified that the route constructed by DCEER does not contain loops and that it can be easily deployed for large-scale WSNs in a distributed manner. By simulation, we have shown that our proposed algorithm not only balances the energy consumption among sensor nodes but also extends the network lifetime and consumes less energy than other approaches if the bounded delay is adjusted appropriately. In summary, in view of a rapid increase in deploying delay-sensitive applications in WSNs, we have contributed a simple and distributed algorithm to save energy while satisfying delay requirements. However, as shown, the message overhead is still a polynomial function. Subsequent studies may look into improving the message overhead to a linear function.

References

[1] T. T. Huynh and C. S. Hong, "An energy* delay efficient multi-hop routing scheme for wireless sensor networks," *IEICE Transactions on Information and Systems*, vol. 89, no. 5, pp. 1654–1661, 2006.

[2] X. Zhang and L. Zhang, "Optimizing energy-latency trade-off in wireless sensor networks with mobile element," in *16th IEEE International Conference on Parallel and Distributed Systems (ICPADS)*, 2010, pp. 534–541.

[3] Y. Jin, D. Wei, A. Gluhak, and K. Moessner, "Latency and energy-consumption optimized task allocation in wireless sensor networks," in *IEEE Wireless Communication and Networking Conference*, 2010, pp. 1–6.

[4] R. Cohen and B. Kapchits, "Energy-delay optimization in an asynchronous sensor network with multiple gateways," in *8th IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, 2011, pp. 98–106.

[5] T. T. Huynh, A.-V. Dinh-Duc, and C. H. Tran, "Energy efficient delay-aware routing in multi-tier architecture for wireless sensor networks," in *International Conference on Advanced Technologies for Communications (ATC)*, 2013, pp. 603–608.

[6] T.-T. Huynh, C.-H. Tran, and A.-V. Dinh-Duc, "Delay-energy aware clustering multi-hop routing in wireless sensor networks," in *Information Science and Applications (ICISA)*. Springer, 2016, pp. 31–40.

[7] K. Akkaya and I. Ari, "In-network data aggregation in wireless sensor networks," in *Handbook of Computer Networks: LANs, MANs, WANs, the Internet, and Global, Cellular, and Wireless Networks*. Wiley Online Library, 2007, vol. 2, pp. 1131–1146.

[8] O. Boyinbode, H. Le, A. Mbogho, M. Takizawa, and R. Poliah, "A survey on clustering algorithms for wireless sensor networks," in *13th International Conference on Network-Based Information Systems (NBiS)*, 2010, pp. 358–364.

[9] S. Rani and S. H. Ahmed, *Multi-hop Routing in Wireless Sensor Networks: An Overview, Taxonomy, and Research Challenges*. Springer, 2016, ch. 2, pp. 15–28.

[10] O. Younis and S. Fahmy, "HEED: A hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks," *IEEE Transactions on Mobile Computing*, vol. 3, no. 4, pp. 366–379, 2004.

[11] K. Akkaya and M. Younis, "Energy and QoS aware routing in wireless sensor networks," *Cluster Computing*, vol. 8, no. 2-3, pp. 179–188, 2005.

[12] Q. Liang, D. Yao, and H. Liu, "Energy-Efficient and Delay-Constrained Routing for different services in Wireless Sensor Network." *Journal of Convergence Information Technology*, vol. 7, no. 5, pp. 233–243, 2012.

[13] S.-W. Han, I.-S. Jeong, and S.-H. Kang, "Low latency and energy efficient routing tree for wireless sensor networks with multiple mobile sinks," *Journal of Network and Computer Applications*, vol. 36, no. 1, pp. 156–166, 2013.

[14] J. Niu, L. Cheng, Y. Gu, J. Jun, and Q. Zhang, "Minimum-delay and energy-efficient flooding tree in asynchronous low-duty-cycle wireless sensor networks," in *IEEE wireless communications and networking conference (WCNC)*, 2013, pp. 1261–1266.

[15] Q. Nadeem, M. B. Rasheed, N. Javaid, Z. Khan, Y. Maqsood, and A. Din, "M-GEAR: Gateway-based energy-aware multi-hop routing protocol for WSNs," in *8th IEEE International Conference on Broadband and Wireless Computing, Communication and Applications (BWCCA)*, 2013, pp. 164–169.

[16] D. L. Guidoni, F. S. H. Souza, J. Ueyama, and L. A. Villas, "RouT: a routing protocol based on topologies for heterogeneous wireless sensor networks," *IEEE Latin America Transactions*, vol. 12, no. 4, pp. 812–817, 2014.

[17] S. Bai, W. Zhang, G. Xue, J. Tang, and C. Wang, "DEAR: Delay-bounded energy-constrained adaptive routing in wireless sensor networks," in *31st IEEE International Conference on Computer Communications (INFOCOM)*, 2012, pp. 1593–1601.

[18] Y. Yao, Q. Cao, and A. V. Vasilakos, "EDAL: An energy-efficient, delay-aware, and lifetime-balancing data collection protocol for heterogeneous wireless sensor networks," *IEEE/ACM Transactions on Networking*, vol. 23, no. 3, pp. 810–823, 2015.

[19] O. Goldreich, *P, NP, and NP-Completeness: The Basics of Computational Complexity*. Cambridge University Press, 2010.

[20] J. Xu, W. Liu, F. Lang, Y. Zhang, and C. Wang, "Distance measurement model based on RSSI in WSN," *Wireless Sensor Network*, vol. 2, no. 8, pp. 606–611, 2010.

[21] Castalia, *Wireless Sensor Network Simulator (Latest version 3.2)*, https://castalia.forge.nicta.com.au/index.php/en/.

[22] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad hoc wireless networks," in *Mobile Computing*. Springer, 1996, pp. 153–181.

Trong-Thua Huynh received his B.Sc. in Computer Science from Ho Chi Minh city University of Science, and M.Sc. in Communication Engineering from Kyung Hee University, South Korea, in 1999 and 2005, respectively. His current research interests include embedded systems, communication technologies and system optimization in wireless sensor networks. He is currently pursuing his Ph.D. study in Computer Science from Ho Chi Minh city University of Technology, Vietnam.

Cong-Hung Tran received the B.Eng. degree in Electronic and Telecommunication Engineering with first class honors from Ho Chi Minh city University of Technology in Vietnam in 1987. He received the B.Eng. degree in Informatics and Computer Engineering from the same university in 1995. He received the M.Eng. degree in Telecommunications Engineering from Hanoi University of Technology in Vietnam, 1998. He received the Ph.D degree from Hanoi University of technology in Vietnam in 2004. His main research areas are B-ISDN performance parameters and measuring methods, QoS in high speed networks, MPLS. He is currently an associate professor in the Faculty of Information Technology II, Posts and Telecommunication Institute of Technology, Ho Chi Minh city, Vietnam.

Anh-Vu Dinh-Duc is an associate professor at the University of Information Technology, Vietnam National University, Ho Chi Minh city, where he has served as Vice-Rector, R&D and External Relations since 2012. He also leads the UIT-VLSI Design group at the Faculty of Computer Engineering. His research interests include WSN, Design Automation of Embedded Systems, Hardware/Software Verification, VLSI CAD, and Reconfigurable Architectures. Dr. Anh-Vu Dinh-Duc received the Master and Ph.D. degrees in Microelectronics from the Institute National Polytechnique de Grenoble (INPG), France, in 1998 and in 2003, respectively. He currently serves as a program/organizing committee member of several ACM and IEEE conferences. He is a valued member of the IEEE.