Energy-Efficient and Delay-Constrained Routing for different services in Wireless Sensor Network

Qingwei Liang, Daoyuan Yao, Haitao Liu

1 Key Lab of Wireless Sensor Network and Communications, Shanghai Institute of Microsystem and Information Technology, Chinese Academy of Science, Shanghai 200050, China, lw198421@126.com
2 Wuxi SensingNet Industrialization Research Institute, Wuxi 214135, China

Abstract
To design a routing protocol that saves sensor node energy while meeting the needs of different applications has become a research focus in wireless sensor network. In this paper, we propose Energy-Efficient and Delay-Constrained Routing (EEDCR) protocol, in which delay-sensitive packets are transmitted along the shortest path to minimize their end-to-end delay, while others are sent to the next hop which is selected based on local neighbor information, e.g., the remaining energy, the proportion covered by the length of data in the total length of the queue and the depth, which alleviates congestion at hot spots. The simulation results show that EEDCR reduces the average end-to-end delay, balances the energy consumption, and prolongs network lifetime.

Keyword: Energy Efficient; Delay Constrained; WSN; Network Lifetime

1. Introduction

Wireless Sensor Network (WSN) consists of very large number of sensor nodes, which enables reliable monitoring and analyses of the physical environment and is different from the traditional networks, so it attracts considerable amount of research efforts from both industry and academia [1]. Usually, sensor nodes in WSN run on batteries and need to survive for a long time without energy supply and timely attention [2]. Naturally, the active time of these nodes rely strongly on how long their batteries can last [3]. Therefore, power saving is critical for WSN, especially in some application scenarios in which recharging or replacing batteries are impossible [4-5]. It is for these reasons that researchers are focusing on the designing of energy efficient protocols and algorithms for WSN.

At the same time, WSN has a wide-range of applications, including military surveillance, disaster prediction, and environment monitoring, and thus has attracted a lot of attention from researchers in the military, industry and academic fields [6]. However, the needs of the applications are different form each other [7]. For example, in smart home application, we should be altered immediately when the fire happens while the other information, such as temperature and humidity, can be delivered within more delay. So different kinds of data should be transmitted in different ways in order to guarantee the real-time requirement and make use of the network resource efficiently [8].

Besides, with the expansion of the applications in WSN, sensor nodes need to deliver more and more data, which would cause congestion at hot spots [7]. To avoid that, routing protocols should scatter packets to light-loaded paths for multi-path transmission [8] [9].

To resolve the aforesaid problems, we propose a novel routing algorithm called Energy-Efficient and Delay-Constrained Routing (EEDCR) protocol. EEDCR categorizes packet into delay-sensitive packet and non delay-sensitive packet by utilizing the flag in its header (i.e., EEDCR considers all the packets with flags 1 as delay-sensitive while the others with 0 as non delay-sensitive, which is the same with PRTR in [10]). To minimize the delay of the delay-sensitive packets, EEDCR sends them along the shortest path to sink while the other ones are delivered to the next hop which is chosen depending on the remaining energy so as to balance the energy of the whole network and increase the survivability of network. At the same time, EEDCR also takes the proportion covered by the length of data in the total length of the queue into account when making the routing decision, which alleviates
the congestion. Besides, to cut down the end-to-end delay further, queue management, which is presented in Section 4, is used to decrease the queuing delay for the delay-sensitive packets. In this way, EEDCR realizes achieving three goals in one protocol: energy balance, delay guarantee, congestion alleviation.

The remainder of this paper is organized as follows. We start with reporting the related work on real-time transmission and energy efficient routing protocols in Section 2. As we concern the delay bound of the delay-sensitive packets, we model the data transmission in the network and calculate the end-to-end delay bound for a single flow based on the Network Calculus theory in Section 3. Section 4 presents in details the mechanism that EEDCR provides real-time transmission, efficient energy consumption and congestion control. In Section 5 we evaluate integrated performance of EEDCR through simulation experiments on a random deployed network with numerical examples. Finally, we draw conclusions in Section 6.

2. Related Work

Trying to network a large number of sensor nodes with constrained resource and meeting the needs of different applications is a challenging problem [11]. In particular, routing, addressing and support for different classes of service are the primary issues to be talked at the network layer, which attract many researchers’ attention. Many methods have been proposed, including concept of potential field in physics.

In [12], A.Basu et al. utilized the steepest gradient search method to propose a potential based routing paradigm to forwards packets in the direction of maximum positive force. However, due to its huge management overhead it isn’t paid enough attention to.

In order to provide real-time transmission using multi-path routing algorithm, the Potential based Real-Time Routing (PRTR) protocol was proposed in [10], which builds a composite potential field by convex combination of node depth field and queue length field using parameter $\alpha$. But it doesn’t take the remaining energy of next hop sensor nodes into consideration. As a result it uses the shortest path frequently, which leads to energy depletion of the nodes along that path, and in the worst case may lead to network partition.

Stankovic et al. designed SPEED in [13] to provide soft end-to-end delay guarantees for sensory data transfers. Nevertheless, SPEED pays no attention to the energy expended along the selected path while it takes into account the delay caused by channel access mechanisms, which means the path along which data is forwarded in SPEED need not be energy-efficient.

Pothuri et al. investigated the problem of finding energy-efficient paths for delay-constrained data in the WSN in [14], in which a set of paths between source and the Sink nodes are identified and indexed in the increasing order of their energy consumption. After estimating the end-to-end delay along each of those ordered paths, the path with the lowest index that satisfies the delay constraint is selected. Similar to [12], it pays too much for the management overhead.

A novel real time routing protocol with load distribution (RTLD) was proposed by Ahmed et al. in [15], which can deliver packets within their end-to-end deadlines while minimizing the network miss ratio and power. To achieve the real time routing and avoid routing holes problem, RTLD makes optimal forwarding decisions depending on link quality, maximum velocity and remaining power, which experiences a high packet delivery ratio and consumes high power consumption in WSN. However, the sensor nodes need to know the location information of itself and its neighbor nodes.

3. Network Model and Delay Analysis

3.1. Network Model

We assume that there is only one destination (i.e. the Sink) in the network, and the other nodes are the same with each other in initial energy, queue length, communication module etc. All nodes are placed randomly and are fixed after being settled.

3.2 Delay Analysis
Since we consider the delay bound as the first goal of our protocol, we derive the end-to-end delay bound for a single flow routed by EEDCR based on the Network Calculus theory which was presented in [16] in 2001. Network Calculus is a \((\min,+)^{+}\) system theory for deterministic queuing systems which builds on the calculus for network delay. As network calculus is built around the notion of cumulative functions for input and output flows of data, the set of real-valued, non-negative, and wide-sense increasing functions passing through the origin plays a major role [17]:

\[
\mathcal{F} = \{ f(x) \mid \forall s,t \in \mathbb{Z}, \forall s \leq t, 0 = f(0) \leq f(s) \leq f(t) \}
\]

**Definition 1.** (Min-plus Convolution and Deconvolution) The min-plus convolution

\[(f \otimes g)(t) = \inf_{0 \leq s < t} f(t-s) + g(s)\]

\[(f \sharp g)(t) = \sup_{u \geq 0} f(t+u) - g(u)\]

**Definition 2.** (Arrival Curve) Given a flow with input function \( R \) a function \( \alpha \in \mathcal{F} \) is an arrival curve for \( F \) iff

\[R(t) - R(t-s) \leq \alpha(t-s) \iff R \leq R \otimes \alpha\]  

Where \( 0 \leq s \leq t \).

**Definition 3.** (Service Curve) If the service provided by a system \( S \) for a given input function \( E \) results in an output function \( E' \) we say that \( S \) offers a service curve \( \beta \) iff

\[E'(t) \geq E(s) + \beta(t-s) \iff E' \geq E \otimes \beta\]

We can calculate the delay bound \( h(\alpha, \beta) \) for the flow at that node by measuring the maximum horizontal distance between a flow’s arrival curve \( \alpha \) and a node’s service curve \( \beta \):

**Theorem 1.** (Delay Bound) Consider a system \( S \) that offers a service curve \( \beta \). Assume a flow \( F \) traversing the system has an arrival curve \( \alpha \). Then we obtain the delay which is bounded by horizontal deviation:

\[d(t) \leq h(\alpha, \beta) = \sup_{t \geq 0} \left\{ \inf_{\Delta d \geq 0} \left[ \alpha(t) \leq \beta(t + \Delta d) \right] \right\}\]  

One of the strongest results of network calculus (albeit being a simple consequence of the associativity of \( \otimes \)) is the concatenation theorem that enables us to investigate tandems of systems as if they were single systems:

**Theorem 2.** (Concatenation Theorem for Tandem Systems) Consider a flow that traverses a tandem of systems \( S_i \) and \( S_j \). Assume that \( S_i \) offers a service curve \( \beta_i \), \( i = 1, 2 \) to the flow. Then the concatenation of the two systems offers a service curve \( \beta_i \otimes \beta_j \) to the flow.

As a simplification, we assume the arrival curves and service curves as:

1. A flow, which traverses node \( i \) and is regulated by a leaky-bucket regulator with \( \rho_i \) and \( \sigma_i \), is constrained by the arrival curve:

\[\alpha_i(t) = \alpha_{\rho_i,\sigma_i}(t) = \rho_i t + \sigma_i\]

Where \( \rho_i \) denotes the sending rate of node \( i \) and \( \sigma_i \) is the maximum size of all packets gotten by node \( i \), which is the buffer-size of the node in [10]. In fact the factor \( \sigma_i \) means flow’s burst size, so we set it more precisely, and we can get a more accurate delay bound.

2. Node \( i \) offers the service curve for the flow with \( R \) as the rate of physical link and \( l \) as the latency for scheduling flow.

3. \( H \) is the maximum hops for the flow to Sink, and we can get \( \gamma R \geq \sum_{i \in H} \rho_i \), where \( \gamma \) denotes the link utilization rate.

Now let’s compute the delay bound. First, we suppose the upperest delay bound exists, and we set \( D_{\text{max}} \) as the upperest delay bound for the flow traversing any node. So one arrival curve for arriving at node \( h \) is:
\[ \alpha_k(t) = \sum_{i=1}^{n} [\rho_i(t + (h-1)D_{\max}) + \sigma_i] \]

We can get the delay bound using Theorem 1:

\[ D_{\max} = h(\alpha_k(t), \beta_k(t)) = D_{\max}(h-1)\gamma + l + \frac{1}{R} \sum_{i\in H} \sigma_i, \quad (4) \]

If \( \gamma < 1/(h-1) \), we can get \( D_{\max} \) according to \( \gamma R \geq \sum_{i\in H} \rho_i \):

\[ D_{\max} = HD_{\max} = H \frac{l + \frac{1}{R} \sum_{i\in H} \sigma_i}{1 -(H-1)\gamma} \geq H \frac{l + \frac{1}{R} \sum_{i\in H} \sigma_i}{1 -(H-1)\gamma} = H \frac{\sum_{i\in H} \sigma_i + lR}{R - (H-1)\sum_{i\in H} \rho_i}, \quad (5) \]

(5) is the upperest delay bound for a single flow traversing \( H \) nodes.

Now that the delay bound is obtained, we can prove it by mathematical induction, and we will not go into detail on this point here.

Since we set \( \sigma_i \) to the maximum size of all packets instead of the buffer-size of nodes and we take the different arrival curve into account, the result we obtain is better than that in [10].

4. Design of the Routing

EEDCR consists of several features including: queue management, power management, depth management, neighborhood management, and routing management as shown in Fig 1. Queue management in each sensor node determines the location in the queue for the receiving packets according to their delay sensitivity; Sensor node gets the information about the remaining energy from power management; Neighborhood management stores and manages the information of the neighbor nodes which is depended on to make routing decision by routing management; Depth management determines the depth of the node; Routing management chooses the next hop node based on the neighbor table information. Also it makes forwarding decision, neighborhood discovery and routing problem handler. The following section describes in details EEDCR components.

![Figure 1. EEDCR Routing Protocol Architecture](image_url)

4.1 Queue Management

According to equation (5), to reduce the latency of the packet, we can reduce the delay of the queue, which is \( R \) in (5). Once receiving packets, sensor node puts them in different locations in the queue according to their delay sensitivity: non delay-sensitive ones are inserted into the end of the queue while delay-sensitive ones are inserted before all the non ones, and packets with the same flag are ordered in line with their arrival time, which is show in Fig 2. The flag is also used as the priority identifier when sensor node delivers packets. Sensitive ones are transmitted before the others.
Queue management adds information about the queue into the sending packets, which is defined as follows:

\[ L_{sp}(i) = \frac{\text{the length of delay-sensitive packets in queue}}{\text{the length of the queue}} \]

\[ L_{p}(i) = \frac{\text{the length of all packets in queue}}{\text{the length of the queue}} \]

The length of delay-sensitive packets, the length of all packets and the length of the queue are shown in Fig 3:

Figure 3. Information about the queue

In [10], it used \( L_{p}(i) \) as the information about the queue to decide next hop for both types of packets, while we use \( L_{p}(i) \) and \( L_{sp}(i) \) respectively according to the sensitivity of the packets, which is more accurate than that in [10].

In this paper, we adapt 802.15.4 as our MAC protocol, so nodes can get every packet from their all neighbors. Before sending packets, node adds its own information about the queue into the packets, when its neighbors receive packets including the information about queue, neighbors get information, in this way nodes can keep the freshness of the information about the queue of their neighbors.

4.2 Power Management

Nodes add their own information of remaining energy which is gotten from power management to the sending packet before packets are delivered to the next hop. And their neighbor nodes get the latest information about remaining power and make the routing decisions more precisely. The remaining energy information of node \( i \) is: \( E_{pi}(i) = E_{ci}(i)/E_{ini} \), where \( E_{ci}(i) \) denotes the current energy of node \( i \) and \( E_{ini} \) denotes the initial energy, which is the same for all nodes.

4.3 Depth Management

In this paper, the depth is the minimum hops between sensor node and Sink, and the depth of Sink is 0, namely \( D(\text{Sink})=0 \). At the beginning, node determines its depth according to the neighbor node which has the minimum depth. When the topology of the network changes, node updates its depth.
Depending on the information of its new neighbors. In Fig 4, node L sets d+1 as its depth according to node I, which has the minimum depth in its neighbor nodes. When node I is dead, node J updates its depth and sets it d+2 according to the depth of node K, in the same way, node L sets its own depth d+3.

Figure 4. Node updates its depth

Besides, node also needs to add its own depth into the sending packets.

4.4 Neighborhood Management

According to the depth relationship between node and its neighbors, every neighbor node is categorized into parent nodes ($N_{p(i)}$), sibling nodes ($N_{s(i)}$), child nodes ($N_{c(i)}$):

$$N_{p(i)} = \{ k \mid D(k) = D(i) - 1, k \in N(i) \}$$

$$N_{s(i)} = \{ k \mid D(k) = D(i), k \in N(i) \}$$

$$N_{c(i)} = \{ k \mid D(k) = D(i) + 1, k \in N(i) \}$$

Where $N(i)$ is the set of nodes which are able to communicate directly with node $i$.

Before node sends the packet to the next hop, it needs to add the information about queue, the remaining energy and the depth into the packet. When its neighbors get the packet, they get the information of the node and store the information into the neighbor table, which is shown in Fig 5:

Figure 5. Neighbors Table
4.5 Routing Management

To realize real-time transmission, congestion alleviation and energy efficient, EEDCR makes routing decisions depending on the neighbor information gotten from neighborhood management and the delay sensitivity of the packets when sending them to the next hop. The probability of choosing neighbor node \( k \) as the next hop for sending packets by node \( i \) is defined as:

\[
P(k) = \begin{cases} 
\alpha \cdot E_p(k) + \beta \cdot (1 - L_p(k)) + \gamma \cdot \frac{1}{\sqrt{D(k)}} \cdot \sqrt{D(i)}, & k \in N_p(i) \\
\sum_{j \in n_p(i)} (\alpha \cdot E_p(j) + \beta \cdot (1 - L_p(j)) + \gamma \cdot \frac{1}{\sqrt{D(j)}} \cdot \sqrt{D(i)}), & k \in (N_p(i) \cup N_s(i)) \end{cases} \tag{6}
\]

where \( \alpha + \beta + \gamma = 1 \), and node \( i \) chooses the neighbor node which has a larger value of probability than any other neighbor as the next hop.

In the Equation (6), to deliver delay-sensitive packets along the shortest way, EEDCR always chooses the next hop from node’s parent nodes, the depth of which is the same with each other, and that means \( \gamma = 0 \). To find the parent node which can deliver the packets in the shortest time, EEDCR overpasses the energy factor and set \( \alpha = 0 \). So we get \( \beta = 1 \), hence Equation (6) can be rewritten as:

\[
P(k) = \sum_{j \in n_p(i)} (1 - L_p(j)) \cdot \frac{1}{\sqrt{D(j)}}, \quad k \in N_p(i) \tag{7}
\]

In this way, routing management selects the node as the next hop which is in the shortest path and can deliver the packet fastest, and it also alleviates the congestion. In equation (8), we take the proportion covered by the length of delay-sensitive packets in the total length of the queue instead of all packets in [10], which is more precisely and leads to better results in selecting the next hop for delay-sensitive packets. And we will prove it in the simulation section.

For non delay-sensitive packets, routing management pays more attention to balancing the energy consumption and alleviating the congestion, so it selects next hop from parent nodes and sibling nodes in equation (7). As a result, non delay-sensitive packets are delivered to the neighbor nodes with more left energy and less packets, which makes them bypass the hot spots to give way to the delay-sensitive ones and saves energy for the nodes with less energy left.

5. Simulation

In this section, simulation experiments were carried out in Qualnet to evaluate the performance of EEDCR. The protocol of MAC layer in our simulation is 802.15.4, and we adapt the same MAC protocols and energy mode in PRTR. The parameters of simulations are listed in Table 1.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAC</td>
<td>802.15.4</td>
</tr>
<tr>
<td>Initial energy</td>
<td>10J</td>
</tr>
<tr>
<td>Packet size</td>
<td>40 bytes</td>
</tr>
<tr>
<td>Queue size</td>
<td>2000 bytes</td>
</tr>
<tr>
<td>Link Layer Transmission Rate</td>
<td>250Kbps</td>
</tr>
<tr>
<td>Delay-Sensitive Packet Rate</td>
<td>Rds (Variable, packets/s)</td>
</tr>
<tr>
<td>Non Delay-Sensitive Packet Rate</td>
<td>Rnds (Variable, packets/s)</td>
</tr>
<tr>
<td>Simulation time</td>
<td>400 seconds</td>
</tr>
</tbody>
</table>
The network, which is randomly deployed, is formed with N nodes densely distributing over a L(m) x L(m) square area, which is shown in Fig 6. With N and L changing, we can form various environments for both protocols to research them deeply.

![Network Topology when L=1200 and N=200](image)

**Figure 6.** Network Topology when L=1200 and N=200

As the delay of the packets with flag 1 is concerned most, first we compared the impact on delay in different algorithm. The parameters in both protocols for selecting the next hop have no influence on delay-sensitive packets, so we set $\alpha=0.5$ in PRTR and $\alpha=0.4$, $\beta=0.4$, $\gamma=0.2$ in EEDCR. Fig 7 and Fig 8 illustrate average end-to-end delay of delay-sensitive packets varies with the changing of the network situation. It is obvious to see that the performance on delay by two protocols both become worse with the expansion of the network or the rate increasing, but EEDCR is better than PRTR, and the larger the network scale, the greater the gap between two protocols, which is shown in Fig 7. The reason lies in the larger scale, the next hop which is selected randomly from the parent nodes in PRTR is more imprecise. However, EEDCR transmits packets to the node which is selected based Equation (7) and can send packets in the shortest time, so it is more adaptive to the large scale network.

![Average Delay](image)

**Figure 7.** The average delay of different network scale when Rds=1.6, Rnds =0.25

![Average Delay](image)

**Figure 8.** The average delay of different rate when L=1000, N=160 and Rnds =0.25

Second, we compared EEDCR with PRTR for energy consumption. We set Rds = 1.6 packets/s and Rnds =0.25packets/s in different network scale of L=1200, N=200 and L=1500, N=250, and Table 2 gives the various values of parameters.
The comparison of EEDCR with PRTR in the time of dead nodes occurrence is shown in Fig 9 and Fig 10. We can see that at the beginning of the simulation EEDCR and PRTR provide almost the same performance, but the gap between two protocols becomes larger with the simulation time, especially in Fig 10, which is simulated in a larger network scale. So EEDCR consumes lower energy than PRTR, which means EEDCR prolongs the network lifetime and makes the energy consumption in all nodes more uniform. The main reason is EEDCR takes the remaining energy into account when selecting the next hop to deliver the non delay-sensitive packets.

Table 2. The values of parameters of the protocols

<table>
<thead>
<tr>
<th>Application</th>
<th>EEDCR (α, β, γ)</th>
<th>PRTR (α)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application 1</td>
<td>0.4, 0.4, 0.2</td>
<td>0.5</td>
</tr>
<tr>
<td>Application 2</td>
<td>0.2, 0.7, 0.1</td>
<td>0.9</td>
</tr>
</tbody>
</table>

Finally, we compare EEDCR with PRTR in throughput ratio of delay-sensitive packets. We set \( R_{ds} = 1.6 \) packets/s and \( R_{nds} = 0.25 \) packets/s in different network scale. The statistics indicate that EEDCR can alleviate congestion by caching non delay-sensitive packets in the idle paths for multi-path transmission to give way to delay-sensitive ones, which can prevent non delay-sensitive ones from being dropped and promote the overall throughput spontaneously. Table 3 and Table 4 prove that EEDCR provides better performance in different situation than PRTR.

Table 3. Throughput Ratio when \( \alpha = 0.4, \beta = 0.4, \gamma = 0.2 \) in EEDCR and \( \alpha = 0.5 \) in PRTR

<table>
<thead>
<tr>
<th>Network Scale</th>
<th>EEDCR (%)</th>
<th>PRTR (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>L=800, N=80</td>
<td>86.8</td>
<td>82.4</td>
</tr>
<tr>
<td>L=900, N=120</td>
<td>85.5</td>
<td>80.7</td>
</tr>
<tr>
<td>L=1000, N=160</td>
<td>83.3</td>
<td>78.3</td>
</tr>
<tr>
<td>L=1200, N=200</td>
<td>80.1</td>
<td>74.4</td>
</tr>
<tr>
<td>L=1300, N=240</td>
<td>77.3</td>
<td>70.2</td>
</tr>
</tbody>
</table>
Table 4. Throughput Ratio when $\alpha=0.2$, $\beta=0.7$, $\gamma=0.1$ in EEDCR and $\alpha=0.9$ in PRTR

<table>
<thead>
<tr>
<th>Network Scale</th>
<th>EEDCR (%)</th>
<th>PRTR (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>L=800, N=80</td>
<td>90.6</td>
<td>87.7</td>
</tr>
<tr>
<td>L=900, N=120</td>
<td>89.7</td>
<td>86.4</td>
</tr>
<tr>
<td>L=1000, N=160</td>
<td>88.3</td>
<td>84.5</td>
</tr>
<tr>
<td>L=1200, N=200</td>
<td>85.6</td>
<td>81.1</td>
</tr>
<tr>
<td>L=1500, N=240</td>
<td>82.9</td>
<td>78.2</td>
</tr>
</tbody>
</table>

To sum up, EEDCR shows a better quality than PRTR in the aspects of average latency of delay-sensitive packets and the time of dead node occurrence and prolonging the network lifetime. Moreover, this routing protocol applies to large scale network, which will have a great extensibility.

6. Conclusion

In this paper, we presented a new routing protocol that is suitable for real-time transmission and long lifetime network. The idea behind the protocol is handling the packets according to their delay sensitivity: delay-sensitive packets are delivered along the shortest path so as to minimize the transmission latency and alleviate congestion; the others are routed based on the remaining energy and queue information in order to balance the energy consumption of the whole network and avoid the hot spots. Experimental results show that the proposed solution is effective in not only satisfying the transmission latency for the delay-sensitive packets, but also in prolonging network lifetime, when compared to conventional routing solutions.

As we use 802.15.4 as our MAC schedule, which can be improved based on the network layer information, in the future our work focuses on finding a cross-layer approach to design a wireless sensor network, which can use upper information to plan MAC layer’s schedule or access control to avoid collision. As a result, the design will be more energy-efficient and provide better system performance.

7. Acknowledgment

This work is supported by Major National S&T Projects of China(No.2010ZX03006-004) and Major State Basic Research Development Program of China(973 Program)(No.2011CB302906).

8. References