Time-Sensitive Utility-Based Single-Copy Routing in Low-Duty-Cycle Wireless Sensor Networks

Mingjun Xiao, *Member, IEEE*, Jie Wu, *Fellow, IEEE*, and Liusheng Huang, *Member, IEEE*

Abstract—Utility-based routing is a routing scheme based on a special composite utility metric. The existing utility-based routing algorithms have not yet considered the delivery delay, so that they cannot work well in low-duty-cycle wireless sensor networks (WSNs). In this paper, we present a time-sensitive utility model. A successful end-to-end message delivery will obtain a positive benefit, which linearly decreases along with an increasing delivery delay; otherwise, a failed delivery will receive zero benefit. The utility is the benefit minus the total transmission costs, no matter if the message delivery succeeds or fails. Such a utility model is analogous to the postal service in the real world. Under this novel utility model, we design two optimal time-sensitive utility-based routing algorithms for the non-retransmission setting and the retransmission-allowed setting, respectively. In our designs, we derive an iterative formula to compute the expected utility of each message delivery, and we present a binary search method to determine the optimal retransmission times. As a result, the two algorithms can achieve the optimal expected utility for each message delivery, which is the optimal balance among the concerned factors, including benefit, reliability, delay, and cost. The simulation results also prove the significant performances of our proposed algorithms.

Index Terms—Distributed algorithms, duty-cycle wireless sensor networks, reliability, routing, time-sensitive utility

1 Introduction

Wireless sensor networks (WSNs) are usually deployed in unmanned application scenarios, such as military surveillance, biological observation, environmental monitoring, and so on. In order to fulfill long-term tasks, these WSNs are generally operated in a low duty-cycle mode to save the energy consumption [2], which are called low-duty-cycle WSNs. So far, several routing algorithms have been proposed for such WSNs [2], [3], [4]. However, these algorithms just use a simple metric (e.g., delivery delay or delivery ratio) as the optimization objective without distinguishing different message deliveries. As a result, the network resources might be exhausted by unimportant message deliveries, so that they cannot serve more important delivery requests.

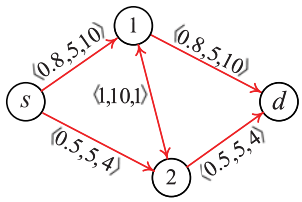
Utility-based routing in traditional unreliable ad hoc networks precisely provides an efficient solution [5], [6]. This is a special routing scheme based on a composite utility metric. A successful message delivery from a source to a destination will obtain a positive benefit as the reward. Otherwise, the failed delivery will receive zero benefit. No matter whether the message delivery succeeds or fails, it will incur a transmission cost. The utility is in terms of the benefit minus the cost. Then, the objective of this routing scheme is to maximize the utility for each message delivery. As a result, such a routing scheme takes the reliability, benefit, and cost into account at the same time, and it can achieve the maximum expected net profit (i.e., benefit minus cost) for each message delivery, which is the optimal balance among the concerned factors [5]. Moreover, an important message delivery in practical applications generally has a large benefit, and a reliable delivery path often charges a large transmission cost. As a result, this routing scheme can inherently deliver an important message to a reliable path, but at a higher cost, and can deliver unimportant messages via those low-cost but unreliable paths, just like the postal service in the real world.

In this paper, we focus on utility-based routing in low-duty-cycle WSNs with unreliable communication links. Compared with traditional ad hoc networks, sensor nodes in low-duty-cycle WSNs periodically schedule themselves to be active for work and then stay dormant at other times to reduce the energy consumption [2], [3], [4]. As a result, each message delivery has a non-negligible delay since it has to wait a certain amount of time until the message receiver becomes active. The delivery delay is thus an important factor for the routing design.

In order to take the delivery delay into account, we introduce time into the utility-based routing model, and propose a time-sensitive utility-based routing (TUR) model. The benefit of a message in this model linearly decreases with the delivery time. The utility is still defined as the benefit minus the transmission cost. Since the benefit is time-related, the delivery delay is indirectly added into the utility

- *M. Xiao and L. Huang are with the School of Computer Science and Technology, Suzhou Institute for Advanced Study, University of Science and Technology of China, Hefei 230027, P.R. China. E-mail: xiaomj, lshuang@ustc.edu.cn.*
- *J. Wu is with the Department of Computer and Information Sciences, Temple University, 1805 N. Broad Street, Philadelphia, PA 19122. E-mail: jiewu@temple.edu.*

Manuscript received 13 Dec. 2013; revised; 23 Apr. 2014; accepted 25 Apr. 2014. Date of publication 29 Apr. 2014; date of current version 8 Apr. 2015. Recommended for acceptance by J. Chen.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TPDS.2014.2321136



benefit	path
50-t	$s \to 1 \to d$
40-t	$s \to 2 \to d$
30-0.1t	$s \to 2 \to 1 \to d$

Fig. 1. An example of time-sensitive utility-based routing on a weighted graph. The edge weight of the graph is $\langle reliability, delay, cost \rangle$. There are three messages with a linearly decreased benefit over time t. The time-sensitive utility-based routing can achieve the maximum utility, i.e., the time-varying benefit minus the cost. Moreover, it will let the three messages be delivered along different paths. Their utility values are calculated in Section 3.1, and are listed in Fig. 6.

model. The objective is still to maximize the utility of each message delivery.

Under this new model, we turn each low-duty-cycle WSN to a weighted graph, and propose a time-sensitive utility-based routing algorithm. The TUR algorithm can maximize the expected utility (the net profit, i.e., the time-varying benefit minus the cost) for each message delivery, which makes the best trade-off among reliability, benefit, delay, and cost. Accordingly, it allows reliability-concerned messages, delay-concerned messages, and cost-concerned messages to be delivered along different paths, as shown in the example of Fig. 1. More specifically, our major contributions include:

1) We present a time-sensitive utility model for low-duty-cycle WSNs. Compared with the existing utility model, the time-sensitive utility simultaneously takes reliability, benefit, delay, and cost into account. As a result, utility-based routing in this model can make a trade-off among the four factors.
2) We propose an optimal time-sensitive utility-based routing algorithm—TUR. The TUR algorithm is a distributed single-copy routing algorithm without retransmission at each hop. In this algorithm, we first derive an iterative formula to compute the expected utility of a given message delivery. Then, this formula is adopted to locally determine the optimal next-hop relay for each node.
3) We also extend our algorithm to cover the case where retransmission is allowed, denoted by TUR-R. For generality, we consider both cases: the retransmission occurs within the same duty-cycle, and at different duty-cycles. In this algorithm, we first derive an upper bound of the optimal retransmission times for each node. Then, a binary search method is proposed to determine the optimal retransmission times.
4) We have conducted extensive simulations to evaluate the TUR and TUR-R algorithms. The results prove that the proposed algorithms can achieve the better expected utility compared to other algorithms. Meanwhile, the results also show that both TUR and TUR-R can make a good balance among reliability, benefit, delay, and cost.

The remainder of the paper is organized as follows. We introduce the low-duty-cycle WSN, the time-sensitive utility model, and the problem of utility-based routing in Section 2. The TUR and TUR-R algorithms are proposed in Sections 3 and 4, respectively. In Section 5, we evaluate the performance of our algorithms through extensive simulations. After reviewing related work in Section 6, we conclude the paper in Section 7.

2 Model and Problem

In this section, we introduce the network model, and the time-sensitive utility model, followed by the problem.

2.1 Network Model

We consider a low-duty-cycle WSN with unreliable communication links. Each sensor only has two possible working states: *the active state*, in which the sensor can perform all the functions of sensing, listening, transmitting, and receiving; and *the dormant state*, in which the sensor turns off all the functional modules except for a wake-up timer. Specifically, when a dormant sensor wakes up, it either switches to the active state, or transmits packets and then switches back to the dormant state. In other words, a sensor can transmit a packet at any time but can receive a packet only when it is active. Before the concrete network model, we first present three reasonable assumptions, which also have been widely adopted in previous works [2], [3], [4].

1) Time is divided into equal-length time slots, and the whole network is loosely synchronized. The synchronization can be achieved through existing approaches, e.g., FTSP [7]. Like previous works [2], [3], [4], a time slot is large enough, so that the time synchronization error can be ignored.

2) Each sensor schedules its working states cyclically. For simplicity, we assume that all sensors share a common duty-cycle and each sensor stays active at only one fixed time slot during each duty-cycle, which is named by the *active time slot* of the sensor. This assumption is reasonable. If sensors have different duty-cycles, the common duty-cycle can be set as their least common multiple. If a sensor has multiple active time slots within a duty-cycle, we can replace this node by several virtual nodes, each of which only has one active time slot in a duty-cycle.

3) The wireless communication links are unreliable, and the CSMA/CA mechanism is adopted to cope with the existence of collision. Previous research shows that the link quality changes very slowly over time [8]. Therefore, the average successful transmission probability derived from history records is adopted to evaluate the link reliability.

Based on the above assumptions, we consider a low-duty-cycle WSN that is composed of a set of sensor nodes, denoted by V. The common duty-cycle is T. If a node locates in the transmission range of another node, we say that they are neighbors. The set of all neighboring nodes of a node i is denoted by N_i. For each pair of neighboring nodes, i and j ($i, j \in V$), there is a successful *transmission probability* $p_{i,j}$. Their active time slots are a_i and a_j ($a_i, a_j \in [1, T]$), respectively. Note that node i gets a message only at the time slot a_i. If it wants to send the message to node j, it must sleep until node j becomes active at the time slot a_j. The transmission delay can be ignored since it is much less than the delay incurred by the sleep. Thus, the message *forwarding delay* from node i to node j is $t_{i,j} = (a_j - a_i) \bmod T$. Besides, the

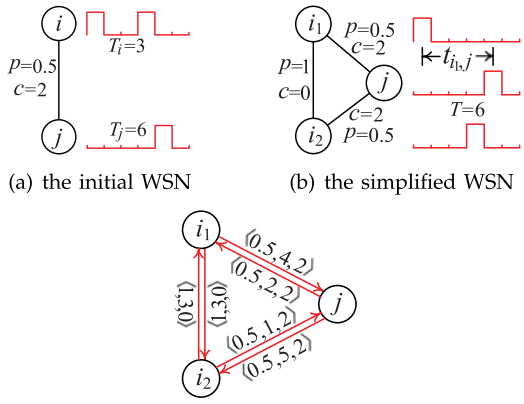


Fig. 2. Example: duty-cycle WSN modeling.

transmission cost from node i to node j is denoted by $c_{i,j}$. Then, we can model the low-duty-cycle WSN as a direct weighted graph $G=\langle V, W\rangle$, where $W=\{\langle p_{i,j}, t_{i,j}, c_{i,j}\rangle | i, j \in V\}$.

Fig. 2 shows an example of low-duty-cycle WSN modeling. Fig. 2a is an initial low-duty-cycle WSN composed of two sensors i and j, whose low-duty-cycles are 3 and 6 time slots, and whose active time slots are 1 and 5, respectively. In Fig. 2b, we utilize two virtual sensors, i_1 and i_2, to replace sensor i. Then, the initial network is simplified to be a low-duty-cycle network, in which there is only one common duty-cycle, and each node only has one active time slot. After computing the delivery delays of neighboring nodes according to their active time slots, we construct the corresponding direct weighted graph, as shown in Fig. 2c. In fact, any low-duty-cycle WSN can be converted to a direct weighted graph in this way.

2.2 Time-Sensitive Utility Model

In the time-sensitive utility model, each message is assigned with a time-varying benefit. When a message is delivered from a source, its benefit will linearly decrease until it reaches its destination. If the benefit becomes zero, this message will be discarded. Each-hop delivery will incur a transmission cost, no matter if the whole message delivery succeeds or fails. The utility is the final benefit minus the total transmission cost, no matter if it is a successful delivery or a failed delivery. More specifically, we define the benefit and utility as follows:

Definition 1. *The* benefit *of a message, denoted as $b(t)$, refers to a linearly decreasing reward over time t if it is successfully delivered to its destination; otherwise, zero reward is returned. Let the* initial benefit *be β, and let the decreased benefit in each time slot be named by the* benefit decay coefficient *and denoted by δ; then the benefit satisfies*

$$b(t)=\begin{cases}\beta - t\cdot\delta, & \text{successful delivery;}\\ 0, & \text{failed delivery.}\end{cases} \tag{1}$$

Here, time t is the living time of the message. A new generated message ($t=0$) has its maximum benefit value. The Time To Live (TTL) of the message is $\frac{\beta}{\delta}$, beyond which the message will be discarded, and the benefit will become zero. Moreover, we let all links share the common benefit decay coefficient. By this way, the delivery delay is linearly combined into the benefit.

Definition 2. *The* utility *of a message delivery, denoted by u, is the benefit minus the total transmission cost of the message delivery, which means the net profit of the message delivery. Let the total transmission cost be c, then the utility satisfies*

$$u = b(t) - c. \tag{2}$$

In this definition, the benefit and the cost are assumed to have been unified as the same unit. Consider a message delivery from a source s to a destination d. If the message successfully arrives at the destination with the delay $t_{s,d}$, the utility would be $b(t_{s,d}) - c$; otherwise if it fails, the utility would be $0 - c$. The utility value is affected by the benefit, the delivery delay, the path reliability, and the transmission cost.

The above notations b, u, and c are related to a whole message delivery from s to d. For simplicity of description, we also define two virtual notions for each node: the remaining benefit of a node and the expected utility of a node. Consider an arbitrary node i in the delivery path from s to d. The remaining benefit and expected utility of node i are defined as follows.

Definition 3. *The* remaining benefit *of node i, denoted by b_i, refers to the remaining benefit value when the message arrives at node i. That is,*

$$b_i = \beta - \delta \cdot t_{s,i}, \tag{3}$$

where $t_{s,i}$ is the total delay for the message being delivered from the source s to node i. Specially, we have $t_{s,s}=0$, and $b_s=\beta$.

Definition 4. *The* expected utility *of node i, denoted by $u_i(b)$, is the expected utility for a message delivery from node i to the destination, in which the remaining benefit of the message is b when it arrives at (or is generated by) node i.*

The two notations b_i and $u_i(b)$ are defined from the point of view of node i, i.e., the case when node i is the current message forwarder. Note that $u_i(b)$ is an expected value. This is because the message delivery from node i to the destination is uncertain. It might succeed or fail at different hops. There are multiple possible results. For each result, there is a probability and a utility value. $u_i(b)$ is the expected value of these utilities. Moreover, $u_i(b)$ is a function of b. A different benefit b will lead to a different expected utility $u_i(b)$.

Fig. 3 illustrates the above concepts through an example. Consider a message delivery from node 1 to a destination d, as shown in Fig. 3a, where the initial benefit is 45, and the benefit decay coefficient is 1. If the message is successfully delivered to the destination d, the final benefit (i.e., the remaining benefit of d) will be $45-5\times 1=40$, and the corresponding utility will be $40-10=30$, as shown in Fig. 3b. If node 1 fails to forward the message to d, the final benefit will be 0, and the corresponding utility will be $0-10=-10$, as shown in Fig. 3c. Thus, the expected utility of node 1 is $u_1(b_1)=0.8\times 30-0.2\times 10=22$.

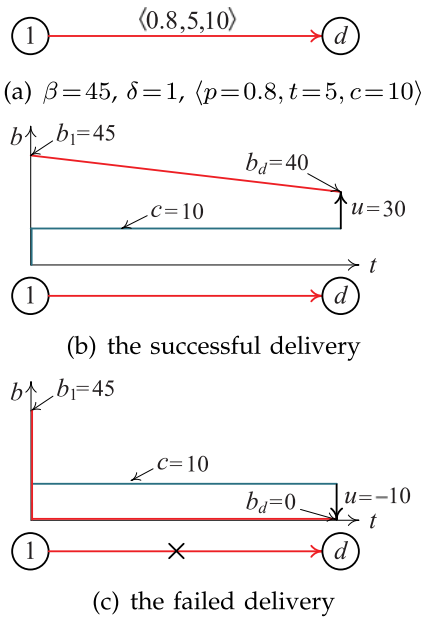


Fig. 3. An example of time-sensitive utility model.

2.3 Problem

Like the previous utility-based routing [5], our objective is to design a routing scheme that can maximize the expected utility of each message delivery under our time-sensitive utility model. Consider a duty-cycle network $G = \langle V, W \rangle$ with a cycle T, as described in Section 2.1, a source node s, a destination node d, an initial benefit β, and a benefit decay coefficient δ. Then, the problem is how to determine the next-hop relay for each node to forward messages, so as to maximize $u_s(\beta)$. For this problem, we take into account two forwarding settings: the non-retransmission setting and the retransmission-allowed setting.

Definition 5. *The* non-retransmission setting *means that, when a node forwards a message to its next-hop relay node, it can only transmit the message once, no matter if the message transmission succeeds or fails.*

Definition 6. *The* retransmission-allowed setting *refers to that, when a node forwards a message, it can transmit the message multiple times, so as to improve the successful probability.*

In this paper, we focus on unreliable WSNs, in which there are *no ACK* messages for each message forwarding. The non-retransmission setting involves the networks with a higher successful transmission probability on average. In contrast, the retransmission-allowed setting is adopted when the average successful transmission probability of a network is not good enough. In addition, we only discuss the solution for a single (s, d, β, δ), which can easily be extended to the case of multiple $(s, d, \beta, \delta)'s$.

For ease of the following presentation, we list the main auxiliary variables in Table 1.

TABLE 1
Description of Major Notations

Variable	Description
T	common duty-cycle of all nodes.
$p_{i,j}$	successful transmission probability between nodes i and j.
$t_{i,j}$	forwarding delay from node i to node j.
$c_{i,j}$	transmission cost from node i to node j.
N_i	neighboring node set of node i.
β	initial benefit (Definition 1).
δ	benefit decay coefficient (Definition 1).
b_i	the remaining benefit of node i (Definition 3).
$u_i(b)$	expected utility for node i to send a message with a remaining benefit b to its destination (Definition 4). Moreover, the expected utility is optimal, unless otherwise stated.
$r_i(b)$	optimal next-hop relay for node i forwarding a message with a remaining benefit b.
$\hat{k}$	optimal single-timeslot retransmission time for the case that retransmissions occur within a duty-cycle.
k^*	optimal retransmission time for the case that retransmissions might occur in one or more duty-cycles.

3 TUR: Non-Retransmission

In this section, we focus on the non-retransmission setting, and propose a distributed time-sensitive utility-based routing algorithm, i.e., TUR, which can achieve the maximum expected utility $u_s(\beta)$ for a message delivery from a source s to a destination d with an initial benefit β and a benefit decay coefficient δ. First, we derive an iterative formula, by which each node can locally compute its expected utility when it knows the expected utility values of neighboring nodes. Second, we present the basic solution, in which the formula is adopted to calculate the optimal expected utility of each node in a distributed manner. Accordingly, the optimal forwarding path is also determined. Finally, we give the detailed algorithm, followed by the analysis on the optimality and convergence.

3.1 The Basic Formula

We first consider an arbitrary delivery path from node s to node d, and derive a formula to compute the expected utility value. Without loss of generality, we let the path be "$s = 0 \to 1 \to \cdots \to n-1 \to d = n$". Then, the expected utility of the message delivery from s to d is $u_s(\beta) = u_0(\beta)$. Assume that all edge weights in the path, including the successful transmission probability, the delivery delay, and the transmission cost, are known. By computing the probability and utility values for each possible delivery case, we can get the formula. More specifically, we have the following theorem.

Theorem 1. *The expected utility value for the message delivery with an initial benefit β and a benefit decay coefficient δ along a given path "$s = 0 \to 1 \to \cdots \to n-1 \to d = n$" satisfies*

$$u_s(\beta) = \prod_{i=0}^{n-1} p_{i,i+1} \left(\beta - \delta \sum_{i=0}^{n-1} t_{i,i+1} \right) - \sum_{i=0}^{n-1} c_{i,i+1} \prod_{j=0}^{i-1} p_{j,j+1}. \quad (4)$$

Proof. We can derive Eq. (4) by computing and summing the utility values of all possible delivery cases.

If the message delivery succeeds, denoted by $s \Rightarrow d$, it means that each-hop message transmission in the path is successful. Then, the delivery delay is the sum

of each-hop delay, i.e., $\sum_{i=0}^{n-1} t_{i,i+1}$. Moreover, the successful delivery probability $P|_{s\Rightarrow d}$, benefit $b|_{s\Rightarrow d}$, and total transmission cost $c|_{s\Rightarrow d}$ satisfy:

$$P|_{s\Rightarrow d} = \prod_{i=0}^{n-1} p_{i,i+1}; b|_{s\Rightarrow d} = \beta - \delta \sum_{i=0}^{n-1} t_{i,i+1}; c|_{s\Rightarrow d} = \sum_{i=0}^{n-1} c_{i,i+1}. \tag{5}$$

If the message delivery fails at the link "$k \rightarrow k+1$" ($0 \le k \le n-1$), denoted by $k \nRightarrow k+1$, the corresponding benefit would become zero, and the total cost only contains the transmission costs for the delivery from s to k. That is,

$$\begin{aligned} P|_{k\nRightarrow k+1} &= (1-p_{k,k+1}) \prod_{i=0}^{k-1} p_{i,i+1};\ b|_{k\nRightarrow k+1} = 0; \\ c|_{k\nRightarrow k+1} &= \sum_{i=0}^{k-1} c_{i,i+1}. \end{aligned} \tag{6}$$

The expected utility u_s is the expected value of the utilities for the successful delivery and all possible failed deliveries. Thus, we have

$$\begin{aligned} u_s(\beta) = {} & P|_{s\Rightarrow d}(b|_{s\Rightarrow d} - c|_{s\Rightarrow d}) \\ & + \sum_{k=0}^{n-1} P|_{k\nRightarrow k+1}(b|_{k\nRightarrow k+1} - c|_{k\nRightarrow k+1}). \end{aligned} \tag{7}$$

Further, after replacing the right side of Eq. (7) by Eqs. (5) and (6) and by combining the related items, we can get Eq. (4). □

Now, we derive an iterative formula which can be used to locally compute the expected utility value. Consider two arbitrary adjacent nodes i and $j = i+1$ ($0 \le i \le n-1$) in the delivery path "$s = 0 \rightarrow 1 \rightarrow \cdots \rightarrow n-1 \rightarrow d = n$". Note that their expected utilities $u_i(b)$ and $u_j(b)$ actually are two functions about the remaining benefit b. For most of the function values, e.g., $u_i(\beta)$ and $u_j(\beta)$, there is not a local iterative relationship between them. Even if the value of $u_j(\beta)$ and the link information between i and j are known, there is no formula that we can use to derive the value $u_i(\beta)$. Fortunately, we find that for a pairwise special remaining benefits b_i and b_j, there is a local relationship between $u_i(b_i)$ and $u_j(b_j)$, as shown in the following theorem.

Theorem 2. *The expected utilities of a node i and its next-hop neighboring node j satisfy:*

$$u_i(b_i) = p_{i,j} u_j(b_j) - c_{i,j}. \tag{8}$$

Proof. We derive the iterative formula about the expected utility values of two neighboring nodes i and j as follows. According to Eq. (4), we get the formulas for $u_i(b_i)$ and $u_j(b_j)$:

$$u_i(b_i) = \prod_{h=i}^{n-1} p_{h,h+1} \left(b_i - \delta \sum_{h=i}^{n-1} t_{h,h+1} \right) - \sum_{h=i}^{n-1} c_{h,h+1} \prod_{g=0}^{h-1} p_{g,g+1}; \tag{9}$$

benefit	β=50, δ=1
direct computation	$u_s(b_s)$=0.8×0.8×(50−1×(5+5))−(10+10×0.8)=7.6
iterative computation	b_s=50, b_1=45, b_d=40
	$u_d(b_d)$= b_d=40
	$u_1(b_1)=P_{1,d}\times u_d(b_d)-c_{1,d}$=0.8×40−10=22
	$u_s(b_s)=P_{s,1}\times u_1(b_1)-c_{s,1}$=0.8×22−10=7.6

Fig. 4. An example of the expected utility computation. The edge weight of the graph is $\langle p, t, c\rangle$. The direct computation and the iterative computation achieve the same result.

$$u_j(b_j) = \prod_{h=j}^{n-1} p_{h,h+1} \left(b_j - \delta \sum_{h=j}^{n-1} t_{h,h+1} \right) - \sum_{h=j}^{n-1} c_{h,h+1} \prod_{g=0}^{h-1} p_{g,g+1}. \tag{10}$$

Comparing $u_i(b_i)$ and $u_j(b_j)$, we have:

$$u_i(b_i) = p_{i,j} u_j(b_j) - \prod_{h=i}^{n-1} p_{h,h+1}(b_i - b_j - \delta \cdot t_{i,j}) - c_{i,j}. \tag{11}$$

Since nodes i and j are adjacent in the delivery path, according to Eq. (3), the remaining benefits of nodes i and j satisfy:

$$b_i = b_j + \delta \cdot t_{i,j}. \tag{12}$$

Therefore, by substituting Eq. (12) into Eq. (11), we can get

$$u_i(b_i) = p_{i,j} u_j(b_j) - c_{i,j}.$$

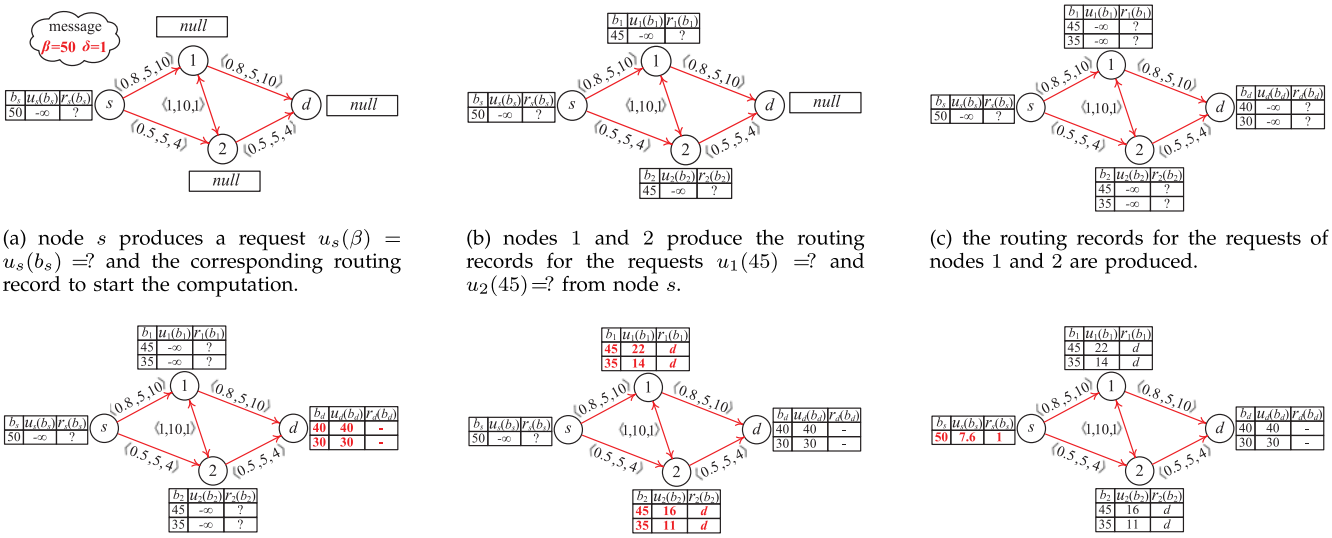
□

Eq. (8) is a local formula, by which each node i can derive its expected utility from that of neighboring nodes. We can also use this formula to iteratively derive the value of $u_s(\beta) = u_s(b_s)$. It will achieve the same result as the direct computation, according to Eq. (4). Fig. 4 shows a simple example, in which the expected utility of the delivery path "$s \rightarrow 1 \rightarrow d$" in Fig. 1 is calculated through the two methods. These results demonstrate that the direct computation and the iterative computation achieve the same result.

3.2 The Basic Solution

The TUR algorithm contains two phases: the initialization phase and the routing phase. In the initialization phase, each node utilizes Eq. (8) to calculate its optimal expected utility. During this computation, the node can determine an optimal next-hop relay. In the routing phase, it just forwards messages via this relay. As a result, the optimal expected utility can be achieved. The detailed method to compute optimal expected utilities and determine optimal relays is presented as follows.

First, the nodes in the network iteratively derive their expected utility values. In order to compute its own expected utility, each node $i \in V - \{d\}$ needs to know the expected utility values of neighboring nodes. Thus, it will send some requests to neighboring nodes for their latest expected utility values. After receiving these expected



(a) node s produces a request $u_s(\beta) = u_s(b_s) =?$ and the corresponding routing record to start the computation.

(b) nodes 1 and 2 produce the routing records for the requests $u_1(45) =?$ and $u_2(45) =?$ from node s.

(c) the routing records for the requests of nodes 1 and 2 are produced.

(d) node d computes $u_d(30)$, $u_d(40)$.

(e) nodes 1 and 2 compute $u_1(b_1)$, $u_2(b_2)$, and determine $r_1(b_1)$ and $r_2(b_2)$.

(f) node s computes $u_s(b_s)$, and determines $r_s(b_s)$.

Fig. 5. Example: compute the expected utility and determine the next-hop relay for each node in Fig. 1.

utility values from neighboring nodes, node i derives its own expected utility. Each node repeatedly requests the latest expected utility values of neighboring nodes to derive its own expected utility, until the iterative process converges. This iterative process is started by the source node producing the request $u_s(\beta) = ?$.

Second, each node $i \in V - \{d\}$ maintains a *routing table*, in which each record is denoted by $\langle b_i, u_i(b_i), r_i(b_i)\rangle$, where $r_i(b_i)$ is the *next-hop relay* selected by node i for it forwarding the messages with the remaining benefit b_i, and this forwarding will achieve an expected utility $u_i(b_i)$. When node i receives a request $u_i(b_i) = ?$ from itself or a neighboring node, it first produces a record $\langle b_i, u_i(b_i) = -\infty, r_i(b_i) = ?\rangle$ in its routing table. Then, it will compute the expected utility and determine the corresponding next-hop relay for this record by using Eq. (8) in Theorem 2. More specifically, node i first calculates the expected utility for the message forwarding via each neighboring node j. Then, it lets the neighboring node, via which the expected utility value is the largest, be its next-hop relay. The corresponding formulas are presented as follows:

$$r_i(b_i) = \underset{j \subseteq N_i}{\operatorname{argmax}} \; p_{i,j} u_j(b_j) - c_{i,j}, \tag{13}$$

$$\text{where } b_j = b_i - \delta \cdot t_{i,j}.$$

$$u_i(b_i) = p_{i,r_i} u_{r_i}(b_{r_i}) - c_{i,r_i}. \tag{14}$$

Third, the destination node d directly computes its expected utility $u_d(b_d)$ for a given remaining benefit b_d, and sends it to the neighboring nodes. The formula to compute $u_d(b_d)$ is presented as follows:

$$u_d(b_d) = b_d. \tag{15}$$

Fig. 5 shows an example to iteratively compute the expected utility and determine the next-hop relay for each node in the network of Fig. 1. At the beginning, the source node s first produces a request $u_s(50) = ?$ for the message delivery ($\beta = 50$ and $\delta = 1$), as shown in Fig. 5a. In order to calculate $u_s(50)$ by using Eqs. (13) and (14), node s needs to know the values of $u_1(45)$ and $u_2(45)$. Then, it sends two requests $u_1(45) = ?$, $u_2(45) = ?$ to nodes 1 and 2, respectively. After receiving the requests, nodes 1 and 2 produce two routing records $\langle b_1 = 45, u_1(b_1) = -\infty, r_1(b_1) = ?\rangle$ and $\langle b_2 = 45, u_2(b_2) = -\infty, r_2(b_2) = ?\rangle$, respectively, as shown in Fig. 5b. Next, nodes 1 and 2 also produce their requests for this message delivery. Then, the corresponding routing records are produced, as shown in Fig. 5c. After the destination node d receives the requests $u_d(40) = ?$ and $u_d(30) = ?$ from nodes 1 and 2, it directly returns $u_d(40) = 40$ and $u_d(30) = 30$ to them. Next, nodes 1 and 2 compute their own expected utility, and determine their next-hop relay, as shown in Fig. 5e. Finally, after receiving the results on $u_1(45)$ and $u_2(45)$, node s also derives its own expected utility $u_s(50) = 7.6$, and selects its next-hop relay as node 1, in Fig. 5f.

3.3 The Detailed Algorithm

Based on our solution, we present the detailed TUR algorithm, as shown in Algorithm 1. In the initialization phase, each node $i \in V$ repeatedly exchanges the expected utility values with its neighboring nodes, and derives its own expected utility and next-hop relay according to Eqs. (13), (14), and (15). More specifically, when node i is active, it first receives the expected utility values of neighboring nodes in Step 5. Then, it calculates its expected utility value and determines the corresponding next-hop relay in Step 6. If the node is the source node, it will first start the iterative computation by producing a request $u_i(b_i) = u_s(\beta) = ?$ in Step 4. When a neighboring node j becomes active, node i will tell node j the latest expected utility that it requires, as shown in Steps 7 and 8. In the routing phase, node i just forwards the message to the selected next-hop relay when it becomes active. The computation overhead of this algorithm is dominated Step 6. In this step, node i calculates its

path \ message	β=50, δ=1	β=40, δ=1	β=30, δ=0.1
$s\to1\to d$	7.6	1.2	0.56
$s\to2\to d$	4	1.5	1.25
$s\to2\to1\to d$	2.5	-1.5	1.7
$s\to1\to2\to d$	1.6	-2.4	0.8

Fig. 6. The forwarding paths and the corresponding expected utility values for three message deliveries in the network of Fig. 1.

expected utility via each neighboring node. Since a node in a real WSN only has a few neighboring nodes, the computation overhead is very small. In addition, each node locally computes its expected utility, determines its next-hop relay, and forwards messages. Thus, it is a distributed algorithm.

Algorithm 1 The TUR algorithm

Require: $G=\langle V, W=\{\langle p_{i,j}, t_{i,j}, c_{i,j}\rangle | i,j\in V\}\rangle$, s, d, β, δ.
Ensure: $u_i(b_i)$, $r_i(b_i)$.
For each node i **do**
Initialization:
1: **for** each time slot in T **do**
2: **if** node i is active **then**
3: **if** node $i=s$ **then**
4: Produce a request $u_i(b_i)=u_s(\beta)=?$ to itself;
5: Request and receive $u_j(b_j)$ from node $j\in N_i$;
6: Calculate each $u_i(b_i)$ and determine r_i by using Eqs. 13, 14, and 15, and update the routing table;
7: **if** neighbor j is active **then**
8: Send the latest $u_i(b_i)$ to node j if j needs it;
Routing:
9: **for** each time slot in T **do**
10: **if** neighbor j is active and $r_i(b_i)=j$ **then**
11: Send the message to node j;

Here, for simplicity of description, we let Algorithm 1 only involve one message delivery from a source node s to a destination node d with an initial benefit β and a benefit decay coefficient δ. In fact, when there are multiple messages with different sources, initial benefits, and benefit decay coefficients, they can be delivered in parallel. The overhead will be multiplied by the number of types of different message deliveries. In general, there are only a few types of message deliveries in a real WSN. Thus, the overhead is acceptable, and the algorithm can still work well. Here, we also ignore the effects of changing link quality on the algorithm. In fact, if the accumulative effect of changing link quality of a node after a long-time running is not negligible, it only needs to start the iterative process in TUR to update its optimal expected utility and optimal next-hop relay, just like the initialization phase. This is a small scale of iterative computation. Moreover, when a node collapses due to energy depletion, or a new node is added into the network, this iterative process is also launched to update the routing tables of the corresponding nodes. The algorithm still works well.

In addition, we use this algorithm to derive the optimal expected utilities and forwarding paths for three different message deliveries in the example network of Fig. 1. The results are listed in Fig. 6, where the records marked by ovals are the optimal expected utilities, and the corresponding paths are the optimal forwarding paths. For each message delivery, we also list the expected utility values of other forwarding paths for each message delivery, to prove the correctness of our algorithm. These results also show that Algorithm 1 can schedule different messages to be forwarded along different paths, so that it has provided a load balance while maximizing the utility for each message delivery, just like the postal service in the real world. Note that this load balance is a natural result of maximizing the utility. It is similar to a market scheduling, which can provide a good load balance result.

3.4 The Convergency and Optimality

The TUR algorithm derives the expected utility of each node through an iterative computation in the whole network. For the convergency of the iterative computation, we have the following theorem.

Theorem 3. *The iterative computation on the expected utility in TUR will not lead to a loop, and it will converge within at most $|V|$ rounds of computation, where a round means that each pair of neighboring nodes will exchange their expected utilities with each other once.*

Proof. First, we show that the iterative computation on the expected utility of each node will not lead to a loop. Consider an arbitrary node $i\in V-\{d\}$, whose expected utility is calculated by using Eq. (14) (or Eq. (8)). For each neighboring node $j\in N_i$, we have

$$u_i(b_i)=p_{i,j}u_j(b_j)-c_{i,j}<u_j(b_j). \tag{16}$$

This means that a neighboring node j can be selected as the next-hop relay only when its expected utility is larger than that of the node i itself. In other words, the expected utility of each node only depends on the neighbors' expected utilities that are larger than its own. Following such a rule, the iterative computation will not result in a loop.

Next, we show that the iterative process will converge within at most $|V|$ rounds of computation. In fact, there must be at least one node whose expected utility value will converge after each round of iterative computation, and it will not change in the following rounds. In the first round, the destination d calculates its expected utility $u_d(b_d)$. Moreover, it will not change in the following rounds of computation since it is the largest expected utility in the whole network. In the second round, the neighboring nodes of the destination d will get $u_d(b_d)$, after which they will derive their own expected utilities. Among them, there must be a node whose expected utility value is the largest, which is also the second largest expected utility in the whole network. This expected utility only depends on $u_d(b_d)$. Thus, it will not change in the following rounds. In other words, this expected utility value has converged. In the same way, the third largest expected utility will converge after the third round of computation, and so on. In each round, at least one node can determine its expected utility and next-hop relay. Thus, the theorem is correct. □

Here, it should be pointed out that the above iterative computation only involves the expected utility. It does not

include the process of initializing routing tables incurred by utility requests. If we take this process into account, the TUR algorithm will converge after $2|V|$ rounds of computation. Then, based on the convergence of TUR, we have that the total computation and communication overheads of each node are $O(|V|^2)$. Moreover, we can straightforwardly derive the optimality of this algorithm as follows.

Theorem 4. *The TUR algorithm is optimal. That is, each node can get its optimal expected utility and next-hop relay after limited rounds of computation.*

Proof. The TUR algorithm utilizes Eqs. (13) and (14) to locally compute the expected utility of each node, and uses Eq. (15) to calculate the expected utility of the destination. Here, Eq. (15) can directly derive the optimal expected utility of the destination. Moreover, Eqs. (13) and (14) let each node select an optimal neighboring node to maximize its expected utility. Thus, if the expected utilities of neighboring nodes are optimal, the expected utility of this node is also optimal. According to the optimality of the expected utility of the destination and the convergence of the algorithm, we can get that the TUR algorithm can let each node achieve the locally optimal expected utility. Note that, the local optimal expected utility is actually equivalent to the global optimal result according to Theorem 2. That is to say, the expected utility of each node is globally optimal. Accordingly, the next-hop relay of each node is also the best. Therefore, the TUR algorithm is optimal. □

4 TUR-R: Retransmission-Allowed

In unreliable communication WSNs, there is generally *no ACK* for each-hop message forwarding. Each node might transmit every message multiple times, so as to improve the successful probability of the message forwarding, while sacrificing some transmission costs. Taking this case into consideration, we extend our solution to the retransmission-allowed setting in this section.

Besides the computation of expected utility, a key problem in the retransmission-allowed setting is to determine the number of retransmissions. To solve this problem, we first consider the case that the time slot is large enough so that the retransmissions only occur within a single time slot, and we compute the optimal retransmission times for this case. Next, we extend it to the general case, in which the time slot is not necessarily a large time interval, and the retransmissions might occur at different duty-cycles. Then, we present a general method to calculate the optimal retransmission times. Finally, based on this method, we propose a retransmission-allowed time-sensitive utility-based routing algorithm—TUR-R.

4.1 Retransmissions in a Single Active Time Slot

First, we consider the case that the retransmission occurs within a single time slot. In fact, if a retransmission occurs within a single time slot, it will improve the successful delivery probability and will also increase the transmission cost, but it will not result in an increased delivery delay. Consider an arbitrary node i and its next-hop node j. After k-time retransmissions, the corresponding successful delivery probability becomes $1-(1-p_{i,j})^k$, and the transmission cost becomes $kc_{i,j}$. Thus, the expected utility for the k-time retransmissions, denoted by $u_i(b_i)|_k$, satisfies the following iterative formula:

$$u_i(b_i)|_k = [1-(1-p_{i,j})^k]u_j(b_j) - kc_{i,j}. \tag{17}$$

Here, Eq. (17) can be seen as a function about the retransmission time k. Moreover, we can find an optimal retransmission time k to maximize the expected utility value $u_i(b_i)|_k$. We call this optimal k the *optimal single-timeslot retransmission time*, and denote it by $\hat{k}$. Regarding $\hat{k}$, we have the following theorem.

Theorem 5. *When node i retransmits a message to its next-hop node j within a single time slot, the expected utility value of node i will decrease after increasing, along with an increase in retransmission times. Moreover, the optimal single-timeslot retransmission time $\hat{k}$ for this message delivery satisfies*

$$\hat{k} = \left\lfloor \frac{\ln c_{i,j} - \ln p_{i,j}u_j(b_j)}{\ln(1-p_{i,j})} \right\rfloor \text{ or } \left\lceil \frac{\ln c_{i,j} - \ln p_{i,j}u_j(b_j)}{\ln(1-p_{i,j})} \right\rceil. \tag{18}$$

Proof. Based on Eq. (17), we compute the expected utility values $u_i(b_i)$ for the k retransmissions and the $k+1$ retransmissions:

$$u_i(b_i)|_{k+1} = [1-(1-p_{i,j})^{k+1}]u_j(b_j) - (k+1)c_{i,j}; \tag{19}$$

$$u_i(b_i)|_k = [1-(1-p_{i,j})^k]u_j(b_j) - kc_{i,j}. \tag{20}$$

With Eqs. (19) and (20), we have

$$u_i(b_i)|_{k+1} - u_i(b_i)|_k = (1-p_{i,j})^k p_{i,j}u_j(b_j) - c_{i,j}. \tag{21}$$

Let k' satisfy $(1-p_{i,j})^{k'}p_{i,j}u_j(b_j) - c_{i,j} = 0$, then we can get

$$k' = \frac{\ln c_{i,j} - \ln p_{i,j}u_j(b_j)}{\ln(1-p_{i,j})}. \tag{22}$$

According to Eq. (21), we have that $u_i(b_i)|_k < u_i(b_i)|_{k+1}$ if and only if $k < k'$. That is, when the number of retransmissions k increases, the expected utility value $u_i(b_i)$ decreases after increasing. Moreover, the maximum expected utility value $u_i(b_i)$ can be achieved only when $k = k'$. Since k is an integer, the optimal single-timeslot retransmission time satisfies

$$\hat{k} = \lfloor k' \rfloor \text{ or } \hat{k} = \lceil k' \rceil.$$

□

We illustrate the relationship between the expected utility $u_i(b_i)|_k$ and the retransmission time k through Fig. 7. Here, $k = k'$ is a real number that maximizes the expected utility $u_i(b_i)|_k$. The optimal single-timeslot retransmission time $\hat{k}$ is the floor of k', which is the largest integer no more than k'.

4.2 Retransmissions in Multiple Active Time Slots

When the time slot is not a large time interval, the retransmissions might occur in multiple active time slots of different duty-cycles. Compared to the single-timeslot retransmissions, the retransmissions in multiple active

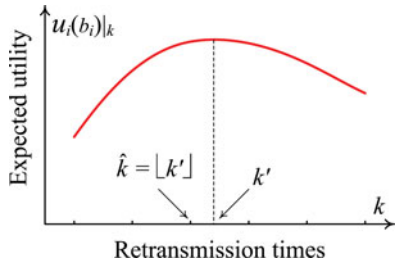


Fig. 7. An example of expected utility for the retransmissions in a single time slot: k' is a real number to maximize the expected utility $u_i(b_i)|_k$, and the optimal single-timeslot retransmission time $\hat{k}$ is the floor of k'.

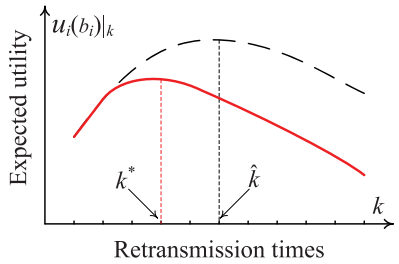


Fig. 8. An example of expected utility for the retransmissions in multiple active time slots: the optimal retransmission time k^* is smaller than the optimal single-timeslot retransmission time $\hat{k}$.

time slots will not only increase the successful delivery probability and the transmission cost, but also will result in a non-negligible delivery delay. Assume that each active time slot can only include μ-times retransmissions. Then, the k-times retransmissions will occupy $\lfloor \frac{k}{\mu} \rfloor$ active time slots, which also means $\lfloor \frac{k}{\mu} \rfloor$ duty cycles.

Consider that an arbitrary node i forwards messages to its next-hop node j through k-times retransmissions. Then, the successful delivery probability of the hth $(1 \leq h \leq k)$ retransmission is $(1-p_{i,j})^{h-1}p_{i,j}$. This retransmission will lead to a delivery delay $\lfloor \frac{h}{\mu} \rfloor T$. Accordingly, the remaining benefit of node j will be decreased by $\delta \lfloor \frac{h}{\mu} \rfloor T$, and the corresponding expected utility of node j is $u_j(b_j - \delta \lfloor \frac{h}{\mu} \rfloor T)$. In addition, the cost of the k-times retransmissions is $kc_{i,j}$. Thus, the iterative formula about the expected utility for the k-times retransmissions becomes:

$$u_i(b_i)|_k = \sum_{h=1}^{k} [(1-p_{i,j})^{h-1}p_{i,j}] u_j \left(b_j - \delta \left\lfloor \frac{h}{\mu} \right\rfloor T \right) - kc_{i,j}. \quad (23)$$

According to Eq. (23), we can still find the optimal retransmission time k to maximize the expected utility value $u_i(b_i)|_k$. We denote this optimal retransmission time by k^*. About k^*, we have the following theorem.

Theorem 6. *When node i forwards a message to its next-hop node j, the expected utility value of node i will decrease after increasing, along with an increase in retransmission times. Moreover, the optimal retransmission time k^* is no larger than the optimal single-timeslot retransmission time $\hat{k}$, i.e., $k^* \leq \hat{k}$.*

Proof. Based on Eq. (23), we compute the expected utility values $u_i(b_i)$ for the $(k+1)$-times retransmissions:

$$u_i(b_i)|_{k+1} = \sum_{h=1}^{k+1} [(1-p_{i,j})^{h-1}p_{i,j}] u_j \left(b_j - \delta \lfloor \frac{h}{\mu} \rfloor T \right) - (k+1)c_{i,j}. \quad (24)$$

With Eqs. (23) and (24), we have

$$u_i(b_i)|_{k+1} - u_i(b_i)|_k = [(1-p_{i,j})^k p_{i,j}] u_j \left(b_j - \delta \lfloor \frac{k+1}{\mu} \rfloor T \right) - c_{i,j}. \quad (25)$$

In Eq. (25), $u_j(b_j - \delta \lfloor \frac{k+1}{\mu} \rfloor T)$ is a decreasing function about the retransmission time k. Thus, there must exist a real number k'' satisfying $u_i(b_i)|_{k''+1} - u_i(b_i)|_{k''} = 0$. Moreover, when the retransmission time $k < k''$, $u_i(b_i)|_{k+1} - u_i(b_i)|_k$ is always larger than zero; otherwise, if $k > k''$, $u_i(b_i)|_{k+1} - u_i(b_i)|_k$ will be smaller than zero. This is to say, when the number of retransmissions k increases, the expected utility value $u_i(b_i)$ decreases after increasing. The optimal expected utility $u_i(b_i)|_k$ can be achieved when $k = k''$. Then, the optimal retransmission time k^* is the closest integer to k'' that can maximize the optimal expected utility $u_i(b_i)|_k$, i.e., $k^* = \lfloor k'' \rfloor$ or $k^* = \lceil k'' \rceil$.

In addition, comparing Eq. (21) and (25), we can derive $k^* \leq \hat{k}$ due to $u_j(b_j - \delta \lfloor \frac{k+1}{\mu} \rfloor T) < u_j(b_j)$. □

Fig. 8 shows an example of the expected utility for the retransmissions in multiple active time slots. The solid line demonstrates the relationship between the expected utility $u_i(b_i)|_k$ and the retransmission time k. $k = k^*$ is the optimal retransmission time. For comparison, we also use a dashed line to show the expected utility for the retransmissions in a single active time slot. Here, the dashed line is just a reference (since the retransmissions in this case occur in multiple time slots). As shown in this figure, when the retransmission time k increases, the expected utility value decreases after increasing, and the optimal retransmission time k^* is smaller than the optimal single-timeslot retransmission time $\hat{k}$.

4.3 The Detailed Algorithm

Based on the above analysis about retransmissions, we can determine the optimal retransmission time k^*. First, we directly compute the optimal single-timeslot retransmission time $\hat{k}$ by using Eq. (18) in Theorem 5. Then, we adopt a binary search in the range $[1, \hat{k}]$ to find the optimal retransmission time k^*. Specifically, we treat the expected utility $u_i(b_i)|_k$ as a discrete function about the retransmission time k, i.e., $U(k) = u_i(b_i)|_k$. Moreover, we use the difference function of expected utility, i.e., $\Delta U(k) = u_i(b_i)|_{k+1} - u_i(b_i)|_k$, to determine the range of binary search. According to Theorem 6, the expected utility is a convex function. Thus, the optimal retransmission time k^* is always located at the range $[k_{low}, k_{high}]$, where the lower bound k_{low} and the upper bound k_{high} satisfy the constraint $\Delta U(k_{low})\Delta U(k_{high}) < 0$. Accordingly, we equally split the range in each round of binary search, and select the part that satisfies this constraint as the next binary search range, until $k_{low} = k_{high}$. Then, this retransmission time is exactly the optimal one. Theorem 6 ensures the correctness.

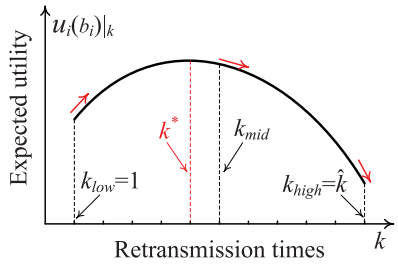


Fig. 9. Searching the optimal retransmission time k^*.

Fig. 9 illustrates this binary search method. At the beginning, we initialize the lower bound and the upper bound as $k_{low} = 1$ and $k_{high} = \hat{k}$, respectively. Additionally, we let the median be $k_{mid} = \lfloor \frac{k_{low}+k_{high}}{2} \rfloor$. Next, we determine which one of the two ranges $[k_{low}, k_{mid}]$ and $[k_{mid}, k_{high}]$ satisfies the above constraint. In Fig. 9, the arrow lines indicate the trends of expected utility. It shows that $\Delta U(k_{low})\Delta U(k_{mid}) < 0$. Thus, the optimal retransmission time k^* must belong to the range $[k_{low}, k_{mid}]$. Then, we select this range as the next binary search range. Repeating this process, we can finally derive the optimal retransmission time k^*.

The detailed process of determining the optimal retransmission times of a given node i is presented in Algorithm 2. In Step 1, node i calculates its optimal single-timeslot retransmission time $\hat{k}$ by using Eq. (18). Then, based on this result, it initializes the search range in Step 2. Next, the binary search is conducted in Steps 3-10. After the search process, node i will get its optimal retransmission times in Step 11. The corresponding computation overhead is $O(\log_2 \hat{k})$.

Algorithm 2 Determine the optimal retransmission times

1: Compute $\hat{k}$ by using Eq. 18;
2: Let $k_{low} = 1$, and $k_{high} = \hat{k}$;
3: **while** $k_{low} \neq k_{high}$ **do**
4: $k_{mid} = \lfloor \frac{k_{low}+k_{high}}{2} \rfloor$;
5: Compute $\Delta U(k_{low}) = u_i(b_i)|_{k_{low}+1} - u_i(b_i)|_{k_{low}}$ by using Eq. 25;
6: Compute $\Delta U(k_{mid}) = u_i(b_i)|_{k_{mid}+1} - u_i(b_i)|_{k_{mid}}$ by using Eq. 25;
7: **if** $\Delta U(k_{low})\Delta U(k_{mid}) < 0$ **then**
8: $k_{high} = k_{mid}$;
9: **else**
10: $k_{low} = k_{mid}$;
11: $k^* = k_{low}$;

Now, we present the retransmission-allowed time-sensitive utility-based routing algorithm, i.e., TUR-R. In fact, by combining the above process of determining the optimal retransmission times and TUR, we can directly get the TUR-R algorithm, as shown in Algorithm 3. Compared to TUR, we add the process of determining the optimal retransmission times at Step 6 in TUR-R. Moreover, each node will compute the expected utility with retransmissions, and will use it to determine its optimal next-hop relay. In addition, the retransmission scheme is adopted by each node in Step 12.

Algorithm 3 The TUR-R algorithm

Require: $G=\langle V, W=\{\langle p_{i,j}, t_{i,j}, c_{i,j}\rangle | i,j \in V\}\rangle$, s, d, β, δ.
Ensure: $u_i(b_i)$, $r_i(b_i)$.
For each node i **do**
Initialization:
1: **for** each time slot in T **do**
2: **if** node i is active **then**
3: **if** node $i = s$ **then**
4: Produce a request $u_i(b_i) = u_s(\beta) = ?$ to itself;
5: Request and receive $u_j(b_j)$ from node $j \in N_i$;
6: Determine the optimal retransmission time k^* and compute each $u_i(b_i)|_{k^*}$ by using Eq. 23 and Algorithm 2;
7: Determine r_i by using Eqs. 13, and update the routing table;
8: **if** neighbor j is active **then**
9: Send the latest $u_i(b_i) = u_i(b_i)|_{k^*}$ to node j if j needs it;
Routing:
10: **for** each time slot in T **do**
11: **if** neighbor j is active and $r_i(b_i) = j$ **then**
12: Send the corresponding messages to node j by k^*-times retransmissions;

Note that, Algorithm 2 and TUR-R are applicable, no matter if the retransmissions occur in a single time slot or multiple active time slots. Moreover, the optimal retransmission times can be determined locally by each node. Therefore, TUR-R can achieve the optimal result in the retransmission-allowed case.

5 Performance Evaluation

In this section, we conduct extensive simulations to evaluate the performances of our proposed algorithms, including TUR and TUR-R. Besides, we also implement three other algorithms to compare them with. The compared algorithms, the evaluation methods, settings, and results are presented as follows.

5.1 Algorithms in Comparison

Since our proposed algorithms are the first time-sensitive utility-based routing algorithms designed for duty-cycle WSNs, to the best of our knowledge, there are no existing algorithms that we can compare them with. Thus, according to the metrics what we are concerned with, we carefully design and implement three other algorithms: *MinDelay*, *MaxRatio*, and *MinCost*.

MinDelay is a shortest-path-based algorithm, in which each node exploits the Dijkstra algorithm to determine the shortest path w.r.t. delay, and then it lets messages be delivered along their shortest paths. MaxRatio lets messages be delivered along the paths which have the largest successful delivery probabilities. MinCost delivers messages along the paths with the smallest expected delivery cost. Both the paths with the largest delivery ratios and the paths with the minimum delivery costs are also determined by the Dijkstra algorithm.

TABLE 2
Evaluation Settings

Parameter name	Default value	Range
Deployment area S	$100m \times 100m$	-
Number of nodes $\|V\|$	-	200-600
Transmission radius	$2.5\sqrt{S/\|V\|}m$	-
Transmission probability	-	0.3-0.9
Transmission cost	-	1-10
Scheduling cycle	20	-
Initial benefit	100	10-100
Benefit decay coefficient	0.02	0.02-0.2
Number of messages	10,000	-

5.2 Simulation Settings and Metrics

In the simulations, we deploy $|V|$ sensor nodes in a 100×100 m square area. More specifically, we divide the whole square area into $|V|$ equivalent small square lattices, and then let each node be deployed at a random position in a lattice. The transmission model of sensor nodes is the traditional disk model. That is, each pair of sensor nodes can communicate with each other only when their distance is less than a given transmission radius. We let all sensor nodes share a common transmission radius, and set the radius to be 2.5 $(> \sqrt{1^2+2^2})$ times the side length of the small square lattice. As a result, the sensor nodes in the neighboring lattices must be within the transmission radius, and thus, can communicate with each other. In this way, the $|V|$ sensor nodes are randomly and uniformly deployed in the whole square area while ensuring that the whole network is fully connected.

Next, we let all of the sensor nodes share a common duty-cycle, and set the cycle to be 20 time slots. Each node becomes active only at one time slot in each cycle. The active time slot is randomly selected while ensuring that it is different from the neighboring nodes'. Each pair of neighboring nodes is associated with a successful transmission probability and cost, which are randomly selected from $[0.3, 0.9]$ and $[1, 10]$, respectively. In addition, the initial benefits and the benefit decay coefficients are selected from $[10, 100]$ and $[0.02, 0.2]$, respectively. All of the evaluation variables are shown in Table 2.

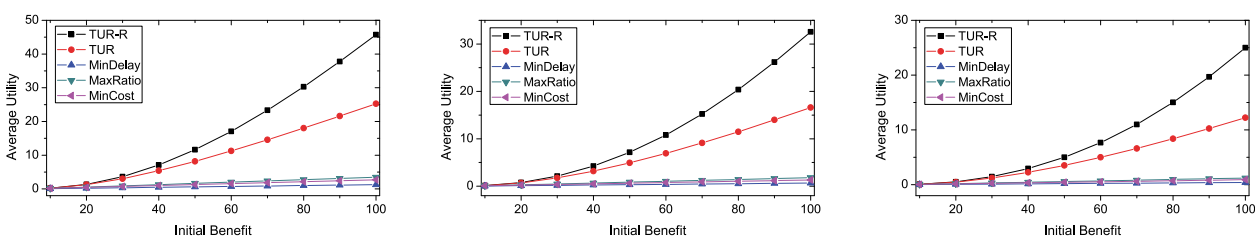
The major metric in our simulations is the *average utility*, which is the average value of utilities of all message deliveries. In order to demonstrate that our utility-based algorithms make a good tradeoff among reliability, delay, and cost, we also compare the *average delivery delay*, *delivery ratio*, and *average delivery cost* of the five algorithms besides the average utility. The average delivery delay and average delivery cost are the average value of delivery delay and the cost of all message deliveries. The delivery ratio is the ratio of successful deliveries and all message deliveries.

5.3 Evaluation Results

We conduct nine groups of simulations in total. In each simulation, we produce $10{,}000$ messages by randomly selecting the sources and destinations, and record the average utility, delivery cost, and delivery delay, respectively. Due to the memory limitation of my computer, we only study the cases $|V| = 200, 400, 600$. Actually, despite this, the evaluation results still show the significant performances of our algorithms. The concrete simulations and results are presented as follows.

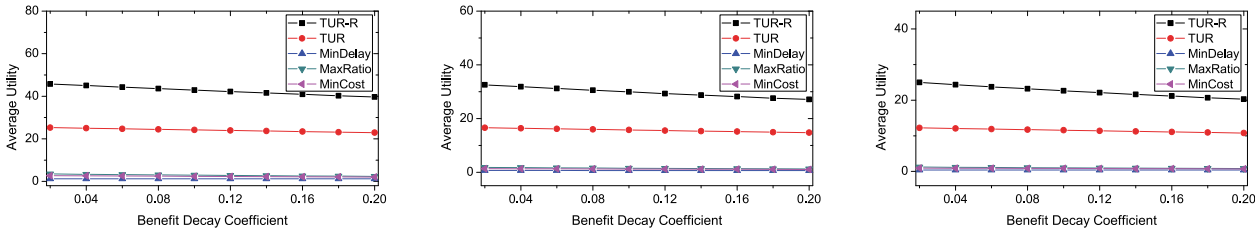
We first evaluate the performance on utility through three groups of simulations. In the first group of simulations, we fix the benefit decay coefficient $\delta = 0.02$ and change the initial benefit value from 10 to 100, i.e., $\beta = 10, 20, \ldots, 100$, to compare the average utility of the five algorithms. The results are shown in Fig. 10. Compared with MinDelay, MaxRatio, and MinCost, TUR increases the utility by 1459.6, 464.3, and 637.3 percent on average, respectively. Compared with TUR, the TUR-R algorithm increases the utility by up to 104.3 percent (47.9 percent on average). In the second group of simulations, we fix the initial benefit $\beta = 100$ and change the benefit decay coefficient from 0.02 to 0.2. The comparison results on the average utility are shown in Fig. 11. Compared with MinDelay, MaxRatio, and MinCost, TUR increases the utility by 2305.2, 923.9, and 1149.9 percent on average, respectively. Compared with TUR, the TUR-R algorithm increases the utility by up to 104.3 percent (87.3 percent on average). In the third group of simulations, we change both the initial benefit and the benefit decay coefficient at the same time to record the change of average utility of the TUR algorithm, as shown in Fig. 12. These results demonstrate the optimal utility performance of our proposed algorithms. Moreover, the larger the initial benefit and the smaller the benefit decay coefficient are, the larger the average utility would be. The results also show that retransmission can achieve an significant increase in performance.

Next, we evaluate the performances on the delivery ratio, delay, and cost through six groups of simulations. We change the initial benefit and the benefit decay coefficient to record the average delivery delay, delivery ratio, and average delivery cost of the five algorithms, respectively. Since the delivery ratios of the five algorithms are different, it is unfair to only compare the average delivery delay and average delivery cost of the successful deliveries. In order to make the comparison fair, we also record



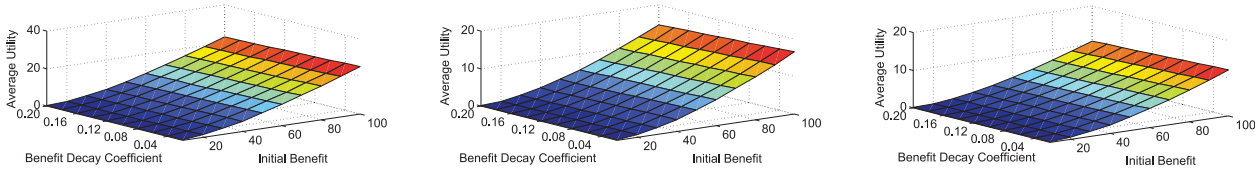
(a) Number of nodes: $|V| = 200$ (b) Number of nodes: $|V| = 400$ (c) Number of nodes: $|V| = 600$

Fig. 10. Performance comparisons of utility versus initial benefit.



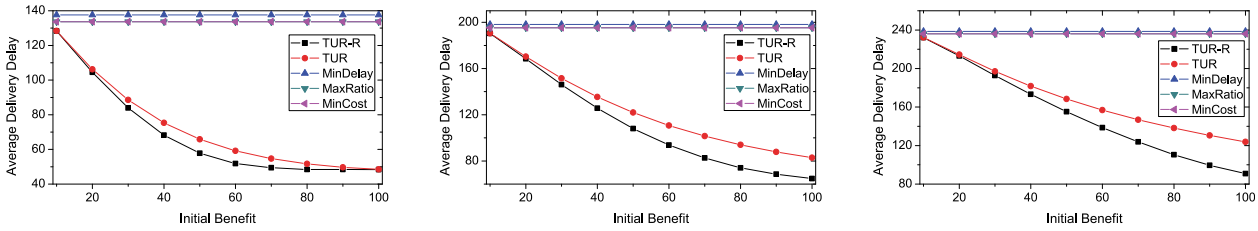
(a) Number of nodes: $|V| = 200$ (b) Number of nodes: $|V| = 400$ (c) Number of nodes: $|V| = 600$

Fig. 11. Performance comparisons of utility versus benefit decay coefficient.



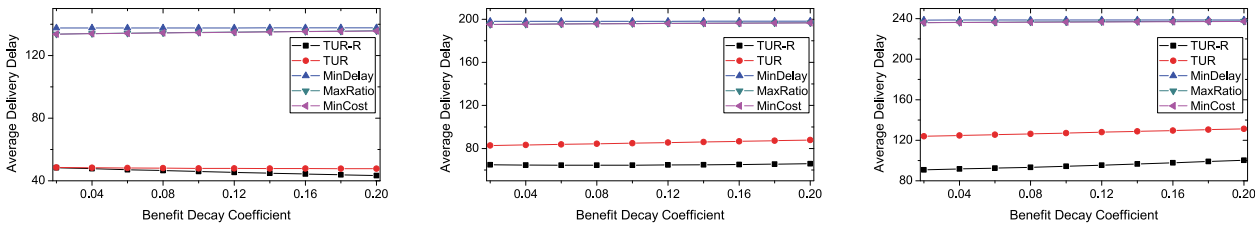
(a) Number of nodes: $|V| = 200$ (b) Number of nodes: $|V| = 400$ (c) Number of nodes: $|V| = 600$

Fig. 12. The relationship of utility versus initial benefit and benefit decay coefficient.



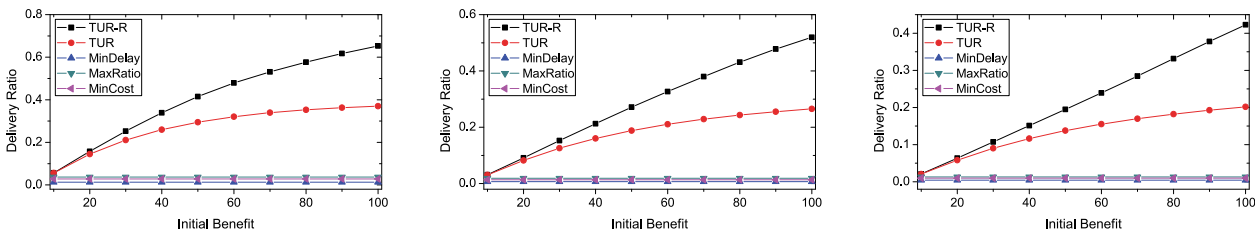
(a) Number of nodes: $|V| = 200$ (b) Number of nodes: $|V| = 400$ (c) Number of nodes: $|V| = 600$

Fig. 13. Performance comparisons of delivery delay versus initial benefit.



(a) Number of nodes: $|V| = 200$ (b) Number of nodes: $|V| = 400$ (c) Number of nodes: $|V| = 600$

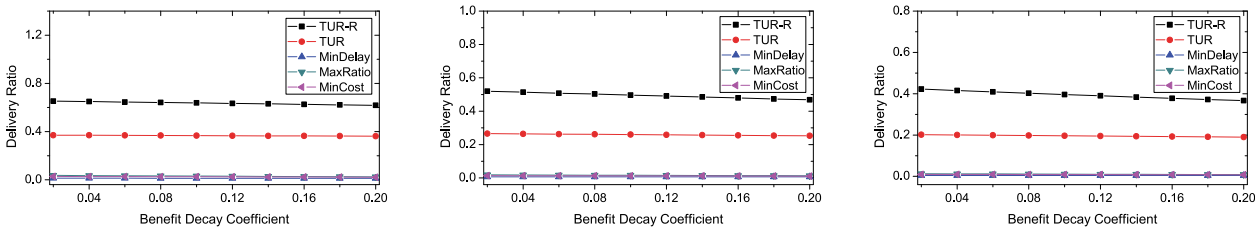
Fig. 14. Performance comparisons of delivery delay versus benefit decay coefficient.



(a) Number of nodes: $|V| = 200$ (b) Number of nodes: $|V| = 400$ (c) Number of nodes: $|V| = 600$

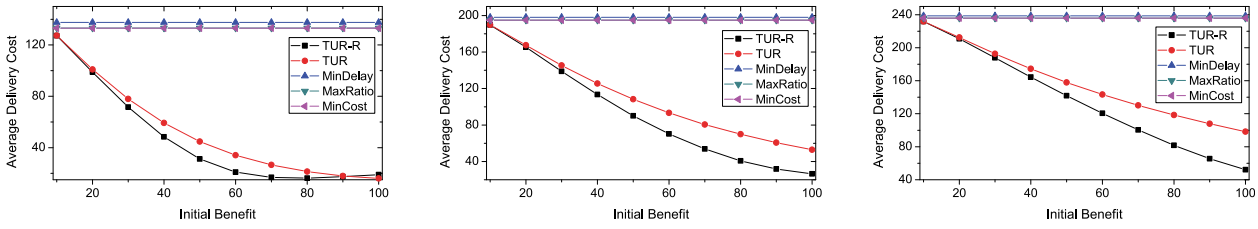
Fig. 15. Performance comparisons of delivery ratio versus initial benefit.

the failed delivery with the maximum delay and cost. The results are shown in Figs. 13-18. Compared with MinDelay, MaxRatio, and MinCost, TUR decreases the delivery delay by 47.0, 46.2, and 46.2 percent on average, increases the delivery ratio by 3096.4, 1144.1, 1184.7 percent, and reduces the delivery cost by 59.1, 58.4, and 58.4 percent, respectively. Here, TUR even has a much better performance with delay and cost than MinDelay and MinCost, due to its good delivery ratio. The results show that the TUR algorithm has achieved good performances with reliability, delay, and cost at the same time. It makes a good tradeoff among the three factors.



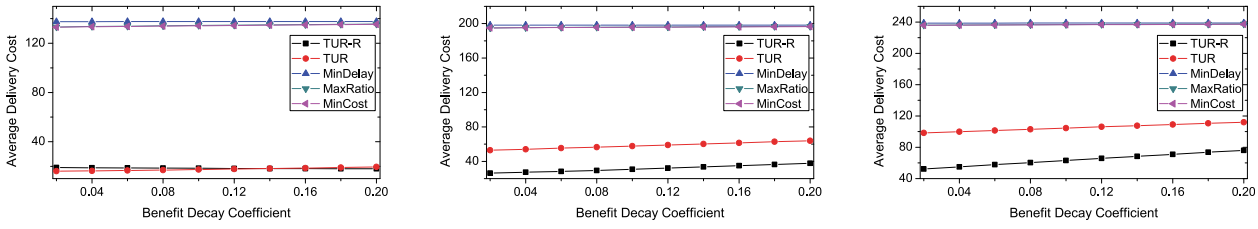
(a) Number of nodes: $|V| = 200$ (b) Number of nodes: $|V| = 400$ (c) Number of nodes: $|V| = 600$

Fig. 16. Performance comparisons of delivery ratio versus benefit decay coefficient.



(a) Number of nodes: $|V| = 200$ (b) Number of nodes: $|V| = 400$ (c) Number of nodes: $|V| = 600$

Fig. 17. Performance comparisons of delivery cost versus initial benefit.



(a) Number of nodes: $|V| = 200$ (b) Number of nodes: $|V| = 400$ (c) Number of nodes: $|V| = 600$

Fig. 18. Performance comparisons of delivery cost versus benefit decay coefficient.

6 Related Work

The routing problem in WSNs has been studied for many years, and a lot of algorithms have been proposed for traditional non-duty-cycle WSNs [9], [10], [11], [12], [13], [14]. In duty-cycle WSNs, the delivery delay is an important factor of routing design. Thus, some delay-concerned routing algorithms, including DSF, L^2, DRINA, etc., [2], [13], [15], [16], [17] and two flooding-based algorithms [3], [4], were proposed recently. However, compared with our utility-based algorithms, none of them adopts the utility metric, which takes benefit, reliability, delay, and cost into account at the same time.

The model of utility-based routing was first proposed by Lu and Wu to balance the reliability and the transmission cost of each message delivery in ad hoc networks [18]. Then, it is extended by adding the opportunistic transmission mechanism in [5], [6]. However, this utility model does not take the delivery delay into account, so that it cannot work well in duty-cycle WSNs.

To this end, we propose the time-sensitive utility-based routing model in [1], [19] by adding the delivery delay into the utility metric. This paper, which takes retransmission into consideration, is exactly the extension of the work in [1]. Moreover, the time-sensitive utility-based routing model in [19], where the failed delivery will lead to zero utility, is actually different from the model in this work. In addition, although the concept "utility" is also adopted widely in other works, it is just a simple composite metric, unlike our utility metric, which is analogous to the postal service in real world [20].

7 Conclusion

In this paper, we present a time-sensitive utility model for duty-cycle WSNs, which takes benefit, reliability, delay, and cost into consideration at the same time. Under this model, we derive an iterative formula to compute the utility of each message delivery. Based on this formula, we design two optimal time-sensitive utility-based routing algorithms for the non-retransmission setting and the retransmission-allowed setting, respectively. Both of the algorithms can maximize the expected utility of each message delivery, and provide a good tradeoff among the four concerned factors. Simulations also prove the significant performances of our proposed algorithms.

Acknowledgments

This paper was an extended version of the conference paper [1] published in IEEE SRDS 2012. This research was supported in part by the National Natural Science Foundation of China (NSFC) (Grant No. 61379132, 60803009, 61003044, 61170058), the NSF of Jiangsu Province in China (Grant No. BK20131174, BK2009150); and NSF Grants ECCS 1231461, ECCS 1128209, CNS 1138963, CNS 1065444, and CCF 1028167.

REFERENCES

[1] M. Xiao, J. Wu, and L. Huang, "Time-sensitive utility-based routing in duty-cycle wireless sensor networks with unreliable links," in *Proc. IEEE 31st Symp. Reliable Distrib. Syst.*, 2012, pp. 311–320.

[2] Y. Gu and T. He, "Dynamic switching-based data forwarding for low-duty-cycle wireless sensor networks," *IEEE Trans. Mobile Comput.*, vol. 10, no. 12, pp. 1741–1754, Dec. 2011.

[3] S. Guo, S. M. Kim, T. Zhu, Y. Gu, and T. He, "Correlated flooding in low-duty-cycle wireless sensor networks," in *Proc. IEEE Int. Conf. Netw. Protocols*, 2011, pp. 383–392.

[4] S. Guo, Y. Gu, B. Jiang, and T. He, "Opportunistic flooding in low-duty-cycle wireless sensor networks with unreliable links," in *Proc. ACM 15th Annu. Int. Conf. Mobile Comput. Netw.*, 2009, pp. 133–144.

[5] M. Lu, F. Li, and J. Wu, "Efficient opportunistic routing in utility-based ad hoc networks," *IEEE Trans. Reliability*, vol. 58, no. 3, pp. 485–495, Sep. 2009.

[6] J. Wu, M. Lu, and F. Li, "Utility-based opportunistic routing in multi-hop wireless networks," in *Proc. 28th Int. Conf. Distrib. Comput. Syst.*, 2008, pp. 470–477.

[7] M. Maróti, B. Kusy, G. Simon, and Ákos Lédeczi, "The flooding time synchronization protocol," in *Proc. 2nd Int. Conf. Embedded Netw. Sens. Syst.*, 2004, pp. 39–49.

[8] S. Lin, J. Zhang, G. Zhou, L. Gu, T. He, and J. A. Stankovic, "ATPC: Adaptive transmission power control for wireless sensor networks," in *Proc. 4th Int. Conf. Embedded Netw. Sens. Syst.*, 2006, pp. 223–236.

[9] J. N. Al-Karaki and A. E. Kamal, "Routing techniques in wireless sensor networks: A survey," *IEEE Wireless Commun.*, vol. 11, no. 6, pp. 6–28, Dec. 2004.

[10] A. Boukerche, B. Turgut, N. Aydin, M. Z. Ahmad, L. Boloni, and D. Turgut, "Routing protocols in ad hoc networks: A survey," *Comput. Netw.*, vol. 55, no. 13, pp. 3032–3080, 2011.

[11] K. Akkaya and M. Younis, "A survey on routing protocols for wireless sensor networks," *Ad Hoc Netw.*, vol. 3, no. 3, pp. 325–349, 2005.

[12] S. Shah and B. Beferull-Lozano, "Joint sensor selection and multihop routing for distributed estimation in ad-hoc wireless sensor networks," *IEEE Trans. Signal Process.*, vol. 61, no. 24, pp. 6355–6370, Dec. 2013.

[13] L. A. Villas, A. Boukerche, H. S. Ramos, H. A. F. de Oliveira, R. B. de Araujo, and A. A. F. Loureiro, "DRINA: A lightweight and reliable routing approach for in-network aggregation in wireless sensor networks," *IEEE Trans. Comput.*, vol. 62, no. 4, pp. 676–689, Apr. 2013.

[14] S. Bai, W. Zhang, G. Xue, J. Tang, and C. Wang, "Dear: delay-bounded energy-constrained adaptive routing in wireless sensor networks," in *Proc. IEEE Conf. Comput. Commun.*, 2012, pp. 1593–1601.

[15] Y. Gu and T. He, "Data forwarding in extremely low duty-cycle sensor networks with unreliable communication links," in *Proc. 5th Int. Conf. Embedded Netw. Sens. Syst.*, 2007, pp. 321–334.

[16] Z. Cao, Y. He, and Y. Liu, "l^2: Lazy forwarding in low duty cycle wireless sensor networks," in *Proc. IEEE Conf. Comput. Commun.*, 2012, pp. 1323–1331.

[17] K. Naveen and A. Kumar, "Relay selection for geographical forwarding in sleep-wake cycling wireless sensor networks," *IEEE Trans. Mobile Comput.*, vol. 12, no. 3, pp. 475–488, Mar. 2013.

[18] M. Lu and J. Wu, "Social welfare based routing in ad hoc networks," in *Proc. Int. Conf. Parallel Process.*, 2006, pp. 211–218.

[19] M. Xiao, J. Wu, C. Liu, and L. Huang, "TOUR: Time-sensitive opportunistic utility-based routing in delay tolerant networks," in *Proc. IEEE Conf. Comput. Commun.*, 2013, pp. 2085–2091.

[20] E. Pagani and G. P. Rossi, "Utility-based forwarding: a comparison in different mobility scenarios," in *Proc. 3rd ACM Int. Workshop Mobile Opportunistic Netw.*, 2012, pp. 29–36.

Mingjun Xiao received the PhD degree from the University of Science and Technology of China (USTC) in 2004. During 2012, he was a visiting scholar at Temple University, under the supervision of Dr. Jie Wu. He is an associate professor in the School of Computer Science and Technology at USTC. He was a TPC or reviewer of many conferences and journals. His main research interests include delay tolerant networks and wireless sensor networks. He is a member of the IEEE.

Jie Wu is the chair and a Laura H. Carnell professor in the Department of Computer and Information Sciences at Temple University. Prior to joining Temple University, he was a program director at the National Science Foundation and a distinguished professor at Florida Atlantic University. His current research interests include mobile computing and wireless networks, routing protocols, cloud and green computing, network trust and security, and social network applications. He regularly publishes in scholarly journals, conference proceedings, and books. He serves on several editorial boards, including the *IEEE Transactions on Computers*, *IEEE Transactions on Service Computing*, and the *Journal of Parallel and Distributed Computing*. He was the general chair for the IEEE IPDPS 2008 and the IEEE ICDCS 2013 and a program cochair/chair for the IEEE INFOCOM 2011 and China Computer Federation (CCF) CNCC 2013. Currently, he is the general chair for ACM MobiHoc 2014. He was an IEEE Computer Society distinguished visitor, ACM distinguished speaker, and the chair for the IEEE Technical Committee on Distributed Processing (TCDP). He received the 2011 China Computer Federation Overseas Outstanding Achievement Award. He is a CCF distinguished speaker and a fellow of the IEEE.

Liusheng Huang received the MS degree in computer science from the University of Science and Technology of China (USTC) in 1988. He is a professor in the School of Computer Science and Technology at USTC. He serves on the editorial board of many journals. He has published six books and more than 200 papers. His main research interests include delay tolerant networks and Internet of things. He is a member of the IEEE.