Wireless Netw (2011) 17:1273–1285
DOI 10.1007/s11276-011-0348-5

CAPF: coded anycast packet forwarding for wireless mesh networks

Xiumin Wang · Kui Wu · Jianping Wang · Yinlong Xu

Published online: 17 May 2011

Abstract In wireless mesh networks, delay and reliability are two critical issues in the support of delay-sensitive applications. Due to sleep scheduling designed for energy efficiency, a node along an end-to-end path needs to wait for its next hop to wake up before it can transmit, which incurs extra delay. In addition, because of unreliable wireless communications, a node may not successfully receive the packet even when it is in active mode. In this paper, we propose a coded anycast packet forwarding (CAPF) scheme for both unicast and multicast communications such that the delay can be reduced and the reliability can be improved. We theoretically analyze the impact of nodes' awake probability and the link loss probability on the end-to-end delay and the reliability. A tradeoff between the end-to-end delay and the reliability is also investigated. Simulation results demonstrate that CAPF provides a flexible mechanism to make good delay-reliability tradeoff and is effective to reduce the end-to-end delay and enhance the reliability.

X. Wang · Y. Xu
Department of Computer Science, University of Science and Technology of China, Hefei, China
e-mail: wxiumin2@student.cityu.edu.hk

Y. Xu
e-mail: ylxu@ustc.edu.cn

X. Wang · J. Wang
Department of Computer Science, City University of Hong Kong, Kowloon, Hong Kong
e-mail: jianwang@cityu.edu.hk

K. Wu (✉)
Department of Computer Science, University of Victoria, Victoria, Canada
e-mail: wkui@cs.uvic.ca

Keywords Wireless mesh networks · Anycast · Unicast · Multicast · Coding

1 Introduction

Recently, wireless mesh networks (WMNs) have emerged as a promising technology to provide the broadband network services. Compared with infrastructure-based networks, WMNs have advantages such as easy deployment, flexible network architecture, self-configuration, and many more. With the increase in both wireless channel bandwidth and the computational capability of wireless devices, WMNs now can be used to support delay-sensitive applications such as video streaming or interactive gaming. Such delay-sensitive applications require that the data content should be propagated to the destination node(s) in a timely fashion.

The wireless devices in WMNs, however, are usually powered by batteries, and as such energy consumption becomes a critical issue, particularly when low-end devices such as sensors or smartphones are used as mesh nodes. As a common practice to save energy, sleep scheduling (i.e., let the devices go to sleep whenever they become idle) has been broadly used, for example, in wireless sensor networks [1, 2] and DigiMesh [3] or ZigBee [4] based WMNs. While sleep scheduling can save energy, it may incur extra delay because a node along an end-to-end path may need to wait for its next hop to wake up before it can transmit. Such a waiting delay could be intolerable for delay-sensitive applications. In addition, wireless channel is usually unreliable, and packet retransmission to improve reliability can have a negative impact on the end-to-end delay guarantee. To sum up, reducing end-to-end delay and guaranteeing reliable data delivery are two contradicting core challenges in WMNs.

To reduce the end-to-end delay, an interesting method is to transmit packets with *anycast* [5]. Instead of having one designated next hop under the traditional packet forwarding schemes, each intermediate node maintains multiple next hops in its forwarding set. The sending node only needs to wait for any one of the next hops to wake up before it can transmit the packet. The end-to-end delay is thus reduced for the waiting delay at each hop. Nevertheless, existing anycast schemes assume reliable radio channels and ignore packet losses [5].

To deal with the unreliable channels, packet retransmissions based on the feedback from the receiving nodes are usually adopted. In delay-sensitive applications such as multimedia streaming, however, in-time packet delivery is much more important since late packets may be useless. We hence should care more about the end-to-end delay as long as the data content can be correctly delivered with a high probability. Due to this reason, TCP-like end-to-end feedback control for reliable per packet delivery is generally avoided for delay-sensitive applications. Following the similar spirit, per-hop retransmission due to channel errors may be also undesirable.

By allowing multiple next hop nodes to receive packets, anycast opens the good opportunities for reducing end-to-end delay and in the meantime achieves a high packet delivery ratio. Anycast alone, however, lacks a control knob that can be used to tune the balance between the end-to-end delay and the reliability. We are thus motivated to design a good mechanism to enhance reliability while the end-to-end delay is effectively controlled.

In this paper, to reduce the end-to-end delay and improve the reliability, we propose using coded anycast packet forwarding (CAPF) scheme in unreliable WMNs for unicast and multicast. To be specific, instead of designating one determined next hop at each step, any active node in a forwarding set can be the candidate to propagate the packet. A sending node can forward the packet once any one of the next hops in its forwarding set wakes up. In addition, with coding, the destination node can decode the native packets once it receives a certain number of coded packets propagated from the source node.

Recently, coding, e.g., source coding and network coding [6], has received extensive research attention in the networking area, and it has been shown that coding can improve network reliability by reducing the number of packet retransmissions in wireless lossy networks [7–9]. Our work differs from the previous literature in that previous work is mainly focused on the traditional packet forwarding scheme, i.e., a given single path for unicast or a given multicast tree for multicast [9]. In contrast, with anycast packet forwarding scheme, the next hop for each step is not designated but is determined by the stochastic sleep scheduling. Thus, the derivation of the delay and the reliability with coded anycast schemes needs to be reconsidered. The previous literature on anycast [5] has tried to minimize the delay in low duty-cycle wireless network. However, the existing work assumes packets are not coded and wireless channels are reliable [5]. While network coding based opportunistic routing [10, 14] is similar to our work, the underlying routing structure in [10, 14] is different and does not take advantages of anycast. To the best of our knowledge, no previous work has considered the coded anycast packet forwarding schemes for unicast as well as multicast to better trade off the end-to-end delay and the reliability.

The main contributions of our work are summarized as follows.

(1) We study the coded anycast packet forwarding scheme from the aspects of both end-to-end delay and reliability in unreliable WMNs.
(2) We theoretically derive the end-to-end delay and the reliability for the unicast communication with our coded anycast packet forwarding scheme. The simulation results confirm the advantages of coded anycast packet forwarding method compared to other packet forwarding schemes.
(3) We also study the coded anycast packet forwarding scheme in multicast case. The simulation results also confirm its advantages in both end-to-end delay and reliability.

The rest of the paper is organized as follows. We first introduce related work in Sect. 2. Section 3 includes the network model. The coded anycast packet forwarding scheme is presented in Section 4. In Sect. 5, we theoretically analyze the end-to-end delay, segment delay, and the reliability of the proposed scheme. The results for performance evaluation are presented in Sect. 6. Finally, we conclude the paper in Sect. 7.

2 Related work

In this section, we review the related work of anycast packet forwarding schemes and network coding.

Recently, researchers have proposed anycast packet forwarding schemes in WMNs [11] to improve system efficiency. Instead of designating one next hop for each step, anycast packet forwarding scheme exploits the spacial diversity of the wireless medium by maintaining multiple next hops in its forwarding set. With coordination between nodes, the next hop that hears the packet and has the highest priority (determined by some rules) in the forwarding set is selected to forward the packet.

Most of the previous anycast work focuses on the forwarding set selection and next hops' priority assignment.

ExOR [11] selects the forwarding sets and prioritizes the next hop based on the link cost information, e.g., ETX. Zorzi and Rao [12] proposed *GeRaF* scheme, which determines the forwarding set and the next hop's priority according to the geographic positions of nodes.

The delay performance with anycast packet forwarding scheme in low duty-cycle wireless networks has been studied in [5, 12]. The work in [5] proposed a dynamic programming based approach to obtain the minimum end-to-end delay from all the sensor nodes to the sink, by iteratively selecting the forwarding set and assigning the next hops' priorities. However, they did not give the theoretical derivation about the end-to-end delay impacted by the sleep scheduling and unreliable wireless channel. The work in [12] studied the end-to-end delay and the energy consumption of the *GeRaF* scheme. However, their work is based on the assumption that each node's sleep/wake-up state follows a deterministic duty cycle, and they do not use coding to enhance reliability.

Network coding [6] based opportunistic routing has been studied in recent years [10, 14]. With random network coding, opportunistic routing allows all the overhearing next hops in the forwarding set to send their coded packets, which are generated by combining the new received packets with the existing packets in their buffers. However, the network coding based opportunistic routing may not be effective for the delay-sensitive applications because it allows all the overhearing next hops to send their coded packets, which incurs extra delay for scheduling these next hops.

The reliability gain of network coding has been studied recently [7–9]. In unicast, the work in [7] has confirmed that network coding can increase the reliability by reducing the number of transmissions per packet. In multicast, the work in [8, 9] showed that network coding improves network reliability. Ghaderi et al. [9] considered different reliable mechanisms based on ARQ and network coding for a given multicast tree topology, and showed that network coding achieves the best performance in terms of the required number of transmissions. However, none of the work considers anycast packet forwarding scheme.

3 Network model

We consider unicast and multicast in unreliable, sleep-scheduled wireless mesh networks. Such networks cover a large range of applications, including for example acoustic wireless sensor networks for ecosystem monitoring [13] or ZigBee based wireless mesh networks [4]. We target at designing new mechanisms to speed up successful data delivery with a high probability. We make the following assumptions.

- *Network model*. We model the network as a directed graph $G(V, E)$ where V is the set of nodes in the network and E is the set of links between nodes. Node $v_{i'}$ is one of node v_i's neighbors only if $(v_i, v_{i'}) \in E$. Let $N(v_i)$ denote the set of node v_i's neighbors, i.e., $N(v_i) = \{v_{i'} | (v_i, v_{i'}) \in E\}$. For unicast, we assume that s and d_0 are the source node and destination node, respectively. For multicast, let s be the source node and $DS = \{d_1, d_2, \ldots, d_N\}$ be the set of destination nodes.
- *Sleep scheduling*. To save energy, we assume that the network adopts a sleep-wake schedule. We assume that time is slotted, and at the beginning of each time slot, each node falls asleep or wakes up independently. Let p be the probability that node v_i is in active mode at any give time slot. This assumption is very generic since whenever energy is not a concern for the network, we can set $p = 1$ for all $v_i \in G$. We assume that if all nodes in the network are in active mode, the whole network must be connected, i.e., we exclude the case that no path can be found between a given source-destination pair.
- *Source messages*. Due to the computational complexity introduced by coding, we assume that the whole data stream is divided into multiple *segments* and coding operations are performed on the packets within one segment instead of the whole data stream. Let n be the number of native packets within one segment, and let $X^m = \{x_1^m, x_2^m, \ldots, x_n^m\}$ be the set of native packets belonging to the m-th segment.

For easy reference, we list the main notations used in the rest of the paper in Table 1.

4 Coded anycast packet forwarding scheme

4.1 Overview of the CAPF scheme

A deterministic path between a source node and a destination over low duty-cycle wireless mesh networks may have a large delay and a high packet lost rate due to the sleep scheduling and unreliable radio links. In order to utilize the broadcast feature of wireless communication, CAPF employs an *anycast* packet forwarding scheme where each intermediate node along the path from the source node to the destination node maintains multiple next hops in its forwarding set.

To determine the forwarding set at each intermediate node, CAPF employs a *virtual path*, a concept that defines fowarding candidates along the path from the source node to the destination node. Such a virtual path not only identifies the forwarding set but also limits the area for message propagation so that communication overhead is controlled. The detail of virtual path construction will be introduced in Sect. 4.2.

Table 1 Main notations and their descriptions

α	The parameter for adjusting the width of a virtual path
d_r	The destination node r
DS	The set of destination nodes
$f_{k'}^{m}$	The k'-th coded packet having the information of packets belonging to the m-th segment
$F(v_i)$	The forwarding set of node v_i
v_i	Node i in the network
L^r_j	The set of nodes at virtual layer j towards destination r
M	The number of virtual layers for a given source-destination pair (s, d_0), in consideration
$N(v_i)$	The set of neighbors of node v_i
N	The total number of destinations in DS, i.e., $N = \|DS\|$
n	The number of native packets in one segment
s	The source node
p	The probability that a node is in active mode at a given time slot
q	The link loss probability of the lossy channel
X^m	The set of native packets belonged to the m-th segment
t_b	The duration of one time slot
t_c	The transmitting time of one packet
Δt	Time interval between two consecutive packets from a source

After a virtual path is constructed, CAPF can use anycast along the virtual path to speed up packet propagation. It can further use coding, e.g., source coding or network coding, to enhance reliability. The coding and packet forwarding procedures will be introduced in Sect. 4.3.

4.2 Bootstrapping: virtual path construction

In this section, we consider the construction of the virtual path in multicast. The unicast case is just a special case of the multicast case, where there is only one destination node in the destination set DS.

The basic idea of virtual path is to identify forwarding nodes which are not too far away from either the source node or the destination node. The virtual path is constructed during the bootstrapping stage. During this phase, all nodes are required to remain active. Since the bootstrapping is a one-time task, its control overhead and the energy cost are at the second-order consideration. After this bootstrapping stage, nodes can sleep and wake up independently to save energy.

The virtual path between a given pair of source and destination nodes, denoted by s and $d_r \in DS$ respectively, can be constructed with a bi-directional search, which is introduced as follows.

(1) The source node s broadcasts a beacon message to the destination nodes in DS, where the beacon message stores the current time.

(2) When destination $d_r \in D$ receives the first beacon message originated from s, d_r calculates the delay the beacon has experienced from s to d_r, denoted by T_{s,d_r}, and broadcasts another beacon message to s.

(3) For an intermediate node v_i, if $T_{s,v_i} + T_{d_r,v_i} < \alpha T_{s,d_r}$, v_i is considered to belong to the virtual path from s to d_r, where T_{s,v_i} (T_{d_r,v_i}) denotes the delay when $s(d_r)$ sends its beacon until v_i receives the first beacon originated from s (d_r), and α is the parameter that is used to constrain the width of the virtual path. The larger the α value, the wider the virtual path.

Note that, with the above approach, the virtual path from the source s to the destination nodes in DS forms a mesh structure, where a node in the virtual path may act as a forwarder to multiple destinations in DS.

Figure 1 shows an example of the virtual path from the source node s to one destination node d_r. All nodes falling within the dotted area form a virtual path that will be used for message propagation between s and d_r.

After constructing the virtual path, we can form the *virtual layers* along the virtual path as follows.

Definition 1 (*Virtual layers*) Moving in the direction from the source to a destination in DS, the source node s is at layer 0. The neighbors of s that are on the virtual path are at layer 1, and the neighbors of layer-i nodes belonging to the virtual path are at layer $i + 1$, and so on.

For unicast, by using the source node to broadcast a beacon message towards the destination along the virtual path, the *virtual layers* from the source node to the destination node can be formed with the algorithm in [15]. We omit this detail due to its triviality. For multicast, we can unite all virtual paths from the source to the destinations as a "super" virtual path. Then virtual layers from the source node to the destination nodes can be formed by requiring the source node to broadcast a beacon message within the "super" virtual path.

Compared with the layered architecture proposed in [20] where the nodes with the same hop count to the base station are grouped into the same layer, i.e., the source node is at layer 0, the nodes that are one-hop away from the source

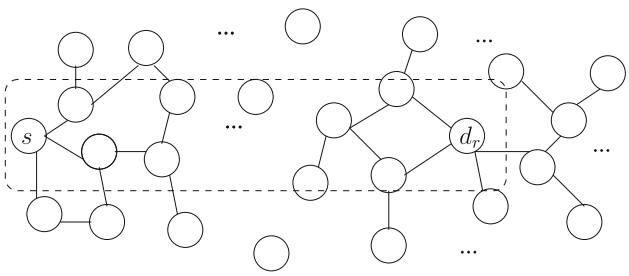

Fig. 1 The virtual path from s to d_r, shown within the *dotted lines*

are at layer 1, and so on, our proposed virtual path/layer construction is more flexible and can identify appropriate forwarding nodes more efficiently. Note that if we set the α to infinity, our method will be the same as that in [20].

4.3 Packet forwarding

(1) *Basic operations*: CAPF uses anycast as the basic message forwarding method. Simply put, the source node s broadcasts a message, and any nodes at virtual layer 1 that are in active mode may receive the message. With node coordination at the same virtual layer, one active node will be selected to broadcast the message to the next virtual layer. Any nodes in the next virtual layer that are in active mode may receive the message. The same process repeats until the destination node(s) receive the message. In the sequel, we will introduce the node coordination at a virtual layer and the different coding schemes to enhance the reliability of the basic anycast scheme.

(2) *Node coordination*: A virtual layer may include multiple active nodes. If all active nodes sends the packet, collision may occur at the receiving nodes at the next virtual layer.

By requiring active nodes at the next virtual layer to periodically send a *live* beacon message, the current sending node can know which neighbors in the forwarding set are active or successfully receive the packets. Then the transmission collision can be resolved by allowing only one active next hop that hears the packet to forward the packet according to some given rules.

As the same as in [5], we assume that the active node that has the highest node ID or highest weight among all active nodes at the same virtual layer is used to forward the message, where the weight of a node is the pre-defined priority value for the node [5]. If network coding is used [6], we can add an extra rule that the active node which has the most number of innovative coded packets among all active nodes at the same virtual layer is allowed to broadcast.

To ease the following description, let $L_j^r = \{v_{j'_1}, v_{j'_2}, \cdots\}$ be the set of nodes that are at virtual layer j towards destination d_r. For an active node v_i at virtual layer j, let $F(v_i)$ be its neighboring nodes that are at virtual layer $j+1$. We call $F(v_i)$ the forwarding set of v_i. Node v_i is allowed to broadcast only if at least one node in $F(v_i)$ wakes up.

4.4 Coded anycast packet forwarding

To enhance the end-to-end reliability, we code the packets transmitted with the anycast packet forwarding scheme. For this purpose, we divide the data stream into segments with each having n packets. As the basic requirement, the coding scheme should allow the destination node to decode a segment after it receives at least n out of $n+k$ coded packets for the same segment.

Different coding schemes, e.g., source coding or network coding, could be employed to fulfill this task.

(1) *Source coding*: If source coding is adopted, the source node can code the original n packets into $n+k$ coded packets and use anycast as described above to transmit the $n+k$ coded packets to the destination. Many source code such as Reed-Solomon (RS) codes [16], Low-Density Parity-Check (LDPC) Codes [17] and random linear codes at the source node can serve the purpose.

(2) *Network coding*: If network coding [6] is used, **At the source node**, since the data stream is divided into segments, random linear coding operations are only performed on packets within the same segment. Let $f_{k'}^m$ be a coded packet which randomly codes the packets belonging to the m-th segment.

At an intermediate active node v_i at virtual layer $j > 0$, suppose that it receives a coded packet $f_{k'}^m$ from another active node at the virtual layer $j-1$. The following procedures will be conducted at node v_i.

- First, node v_i compares the segment sequence of packet $f_{k'}^m$ with the highest segment sequence number it has seen so far.
- If the latest segment sequence that node v_i has seen so far is higher than m, which means that it has already forwarded coded packets belonging to m-th segment, node v_i discards packet $f_{k'}^m$. Otherwise, node v_i will check whether $f_{k'}^m$ is innovative [6] with respect to the m-th segment. If $f_{k'}^m$ is not an innovative packet, node v_i discards the packet as well.
- Otherwise, node v_i first maintains $f_{k'}^m$ in its buffer, and combines the new packet $f_{k'}^m$ with its previously-buffered packets that also belong to the m-th segment into a new coded packet. It waits until at least one of the nodes in its forwarding set wakes up, and then broadcasts the coded packets if the rules regulated in the *node coordination* section are satisfied.

At the destination node d_r, after receiving a coded packet $f_{k'}^m$, it checks whether this coded packet is innovative with respect to the same segment. If yes, node d_r buffers $f_{k'}^m$; otherwise, d_r discards it. After receiving n innovative coded packets belonging to the same segment, node d_r can successfully decode the original packets in the segment.

5 Analysis on reliability and expected end-to-end delay

5.1 Reliability

We first consider unicast from source s to a destination d_0.

Definition 2 Reliability from the source s to a destination d_0, denoted by R_{d_0}, is defined as the probability that the packets in one data segment sent by s can be correctly recovered at the destination d_0.

Let $L^0{}_i$ denote the set of nodes at the virtual layer i and $l_i = |L^0{}_i|$, i.e., l_i represents the number of nodes at the virtual layer i. To simplify presentation, we assume that any node at the virtual layer i can communicate with any node at virtual layers $i-1$ and $i+1$. Without this assumption, analytical results can still be obtained following the same steps in our analysis, but the notation will become complex, since each node would need to keep track of its own forwarding set at the next virtual layer. Without loss of generality, we regard the source node s and destination node d_0 as at the virtual layer 0 and the virtual layer M, respectively.

Let P_i be the probability that a given packet is successfully delivered from the virtual layer i to the virtual layer $i+1$. This probability is equal to the probability that when j $(1 \le j \le l_{i+1})$ nodes wake up at the next virtual layer, at least one node correctly receives the packet. Therefore, we have

$$P_i = \sum_{j=1}^{l_{i+1}} \frac{\binom{l_i+1}{j}}{2^{l_{i+1}-1}}(1-q^j) \tag{1}$$

The probability P that a given packet is correctly received by the destination d_0 can be calculated as:

$$P = \prod_{i=0}^{M-1} P_i \tag{2}$$

We next derive the reliability from the source to the destination. To this end, we assume that the coding scheme possesses the property that once the destination correctly receives at least n coded packets, it can successfully decode the original segment.

To ease analysis, we only consider the case of source coding such that for every segment, the source s sends out $n+k$ coded packets, and the destination can decode the segment as long as it correctly receives at least n coded packets. For the case of network coding, although approximate analysis can be obtained by approximating the path from the source to the destination as a broadcast channel and performing random linear code at the source [19], the exact analysis with network coding is hard to obtain since we need to track innovative packets at each virtual layer. We leave it as an open problem.

To derive R_{d_0}, we need to calculate the probability that the destination successfully receives at least n coded packets after the source s sends out $n+k$ coded packets, which is,

$$R_{d_0} = \sum_{j=n}^{n+k} \binom{n+k}{j} P^j (1-P)^{n+k-j} \tag{3}$$

For multicast, we define different metrics regarding reliability, which can be calculated with the above analytical results.

Definition 3 Average reliability of a multicast group with the source s and the destination set DS, denoted by $Ar(DS)$, is defined as the average value of the reliabilities from the source s to the destination nodes in DS. Formally,

$$Ar(DS) = \frac{1}{N} \sum_{d_r \in DS} R_{d_r}, \tag{4}$$

where $N = |DS|$.

Definition 4 Worst reliability of a multicast group with the source s and the destination set DS, denoted by $Wr(DS)$, is defined as the minimum value of the reliabilities from the source s to the destination nodes in DS. Formally,

$$Wr(DS) = \min_{d_r \in DS} R_{d_r} \tag{5}$$

5.2 Expected end-to-end packet delay

We next consider the expected end-to-end packet delay from the source s to a given destination d_0. Following the same assumption as above and letting D_i denote the delay the packet experienced at the virtual layer $i, i = 0, 1, \ldots, M-1$. We have

$$E[D_i] = \sum_{h=0}^{\infty} (ht_b + t_c)(1-p)^{hl_{i+1}} \left(1 - (1-p)^{l_{i+1}}\right) \tag{6}$$

The first item $ht_b + t_c$ represents the waiting time of the packet at the virtual layer i plus the time for transmitting the packet. The second item $(1-p)^{hl_{i+1}}$ represents the probability that none of the nodes at the next virtual layer is active for up to h time slots, and the last item $(1-(1-p)^{l_{i+1}})$ is the probability that at least one node at the next virtual layer is active at $(h+1)$-th time slot, which is the condition that the packet can be forwarded to the next layer.

Since the end-to-end delay $D = \sum_{i=0}^{M-1} D_i$, by the linearity of expectation, we have the expected end-to-end packet delay

$$
\begin{aligned}
E[D] &= \sum_{i=0}^{M-1} E[D_i] \\
&= \sum_{i=0}^{M-1} \sum_{h=0}^{\infty} (ht_b + t_c)(1-p)^{hl_{i+1}} \left(1 - (1-p)^{l_{i+1}}\right) \\
&= \sum_{i=0}^{M-1} \left(\frac{t_b (1-p)^{l_{i+1}}}{1 - (1-p)^{l_{i+1}}} + t_c \right) \\
&= t_b \sum_{i=0}^{M-1} \frac{(1-p)^{l_{i+1}}}{1 - (1-p)^{l_{i+1}}} + M \times t_c
\end{aligned}
\tag{7}
$$

Note that $E[D]$ is meaningful only when the packet is successfully delivered from the source to the destination.

5.3 Expected end-to-end segment delay

We then derive the expected end-to-end segment delay from the source s to a given destination d_0, i.e., *the expected delay from s sends out the first coded packet in a segment until the destination d_0 recovers all n packets in the segment*.

Let Δt be the time interval between two consecutive packets sent out from s. To help understanding, Fig. 2 illustrates the time axis for the packet propagation. For example, if the n-th coded packet received at the destination d_0 is the j-th packet sent from s where $j \geq n$, then the end-to-end segment delay is $(j-1)\Delta t + D$, where D is the end-to-end packet delay whose expectation can be calculated with Eq. 7.

Let P'_j be the probability that the n-th coded packet received at the destination is the j-th packet ($j \geq n$) sent from the source. In other words, the destination has successfully received $n-1$ packets out of the first $j-1$ packets sent from the source, and it also successfully receives the j-th packet. It is easy to obtain:

$$
\begin{aligned}
P'_j &= \binom{j-1}{n-1} P^{n-1} (1-P)^{(j-1)-(n-1)} P \\
&= \binom{j-1}{n-1} P^n (1-P)^{j-n},
\end{aligned}
\tag{8}
$$

where $j \geq n$ and P can be calculated with Eq. 2. The term $\binom{j-1}{n-1} P^{n-1}(1-P)^{(j-1)-(n-1)}$ denotes the probability that the destination has successfully received $n-1$ packets out of $j-1$ packets, and the last term P denotes the probability that the destination successfully receives the j-th packet.

Let S_{d_0} be the end-to-end segment delay from the source s to the destination d_0. We therefore have:

$$
E[S_{d_0}] = \sum_{j=n}^{n+k} (j-1)\Delta t P'_j + E[D] \tag{9}
$$

Note that similar to the expected end-to-end packet delay, $E(S_{d_0})$ is meaningful only when at least n coded packets for a segment are received by the destination d_0.

For multicast, we define different metrics to evaluate segment delay, which can be calculated with above analytical results.

Definition 5 Average segment delay of a multicast group with the source s and the destination set DS, denoted by $Asd(DS)$, is defined as the average value of the expected end-to-end segment delay from s to the destination nodes in DS. Formally,

$$
Asd(DS) = \frac{1}{N} \sum_{d_r \in DS} E[S_{d_r}] \tag{10}
$$

where $N = |DS|$.

Definition 6 Worst segment delay of a multicast group with the source s and the destination set DS, denoted by $Wsd(DS)$, is defined as the maximum of the expected end-to-end segment delay from s to the destination nodes in DS. Formally,

$$
Wsd(DS) = \max_{d_r \in DS} \{E[S_{d_r}]\} \tag{11}
$$

5.4 Tradeoff between reliability and expected end-to-end segment delay

Clearly, both the reliability and the expected end-to-end delay depend on k, which is determined by the coding

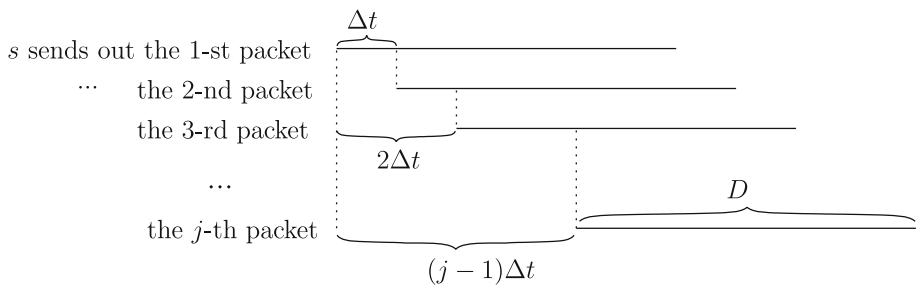

Fig. 2 The time axis of packet propagation

scheme. By adjusting the k value, we thus obtain a mechanism to make tradeoff between the reliability and the end-to-end delay.

As one example, if a given reliability R_{d_0} is required, i.e, $R_{d\_0} \geq \beta$, we can minimize the expected end-to-end segment delay $E[S_{d\_0}]$ as follows:

$$\min_k E[S_{d_0}]$$

subject to:

$$R_{d_0} \geq \beta$$
$$k > 0,$$

where β is the given threshold to constrain reliability, i.e., $0 < \beta \leq 1$.

As another example, if we constrain the expected end-to-end segment delay $E[S_{d_0}]$, the reliability R_{d_0} can be maximized as follows:

$$\max_k R_{d_0}$$

subject to

$$E[S_{d_0}] \leq \gamma$$
$$k > 0,$$

where γ is the given threshold for the expected end-to-end segment delay.

6 Performance evaluation

In this section, we demonstrate the effectiveness of our CAPF scheme through simulations. In our simulations, we generate a connected network, in which 80 nodes are randomly deployed in a two-dimensional (2D) space. The source node and the destination node(s) are random selected. We divide the whole data content to be sent into multiple segments where each segment has n packets. After the bootstrapping stage, at each time slot, each node is in active mode with the probability of p. Link loss probability is set to be q.

To demonstrate the advantage of CAPF, we introduce five baseline schemes, namely, *optimal anycast scheme, anycast scheme, coded opportunistic scheme, coding scheme*, and *no-anycast-no-coding scheme*. We briefly introduce how these four schemes operate in unicast and multicast cases.

- *Optimal anycast scheme*. The optimal anycast scheme proposed in [5] is an anycast based algorithm, which selects the forwarding set and prioritizes the next hops in the forwarding set by a dynamic programming based approach to minimize the end-to-end delay. Since this scheme is only used for unicast, we will only compare it with CAPF for unicast.
- *Anycast scheme*. The only difference between this scheme and our CAPF scheme is that the packets propagated from the source node to the destination(s) are not coded. The path used in this scheme is the same as in our CAPF scheme.
- *Coded opportunistic scheme*. This scheme is proposed in [14], which allows multiple next hops in the forwarding set to encode the received packets and send them out.
- *Coding scheme*. This scheme does not use anycast. For the case of unicast, the Dijkstra algorithm is used to find the shortest path from the source node to the destination node. For the case of multicast, the algorithm proposed in [18] is used to find the minimum cost multicast tree, where the cost of the link is defined as the hops. In addition, the source coding operation as in the CAPF scheme is conducted.
- *No-anycast-no-coding scheme*. This scheme constructs the routing path using the same approach as in the above *coding scheme*. However, the packets propagated from the source to the destination(s) are not coded.

Each simulation result is based on 120 simulation instances, and is presented with 95% confidence interval.

6.1 The comparison of segment delay between simulation results and analytical results

We first demonstrate the effectiveness of the proposed CAPF scheme with source coding by comparing the segment delay obtained through simulation with the analytical results. We vary the threshold α in the range of 1.0, 3.0 for $k = 30, p = 0.8, q = 0.1, t_b = 1, t_c = 2, \Delta t = 5$.

In the first setting, we evaluate the segment delay in unicast under two cases: $n = 10$ and $n = 20$. As shown in Fig. 3, the segment delay obtained by our simulation is

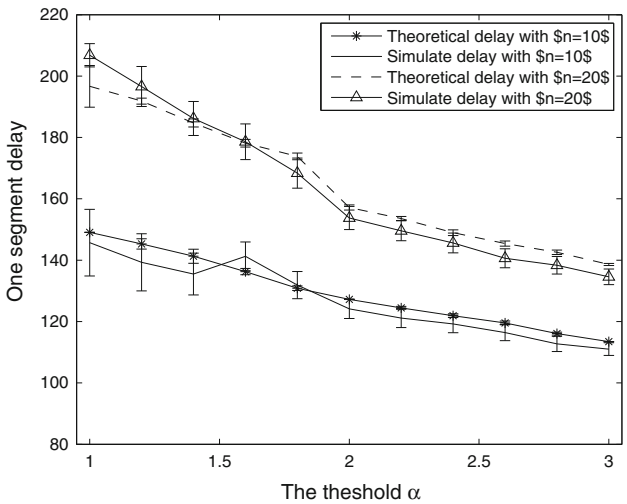

Fig. 3 The segment delay versus the threshold α for unicast

very close to the analytical result. We can also see that the segment delay decreases with the increase of α, because the virtual path becomes wider and thus the number of candidate forwarders at each virtual layer becomes larger, which decreases the waiting delay at each hop. In addition, with the increase of n, the delay for one segment data increases since the number of native packets needed to send in one segment increases.

In the second setting, we evaluate the segment delay in multicast by setting $N = 10$, $n = 10$. Figure 4 gives both the worst segment delay and the average segment delay. As in unicast case, the segment delay obtained by our simulation is also close to the analytical results in multicast, and the segment delay decreases with the increase of α.

6.2 The comparison of reliability between simulation results and analytical results

We then evaluate the reliability of the CAPF scheme with source coding by comparing the simulation results with the analytical results. We vary the link loss probability q in the range of [0.0, 0.30] for $\alpha = 1.5$, $n = 10$.

In the first setting, we evaluate the reliability in unicast under two cases: $k = 10$ and $k = 15$. As shown in Fig. 5, the reliability obtained by the simulation is close to the analytical result. We also can see that the reliability decreases with the increase of the link loss probability q, because the packet will be lost with higher probability. In addition, the reliability with a larger k value is higher than that with a smaller k value. This is because the number of coded packets originated from s with a larger k is more than that with a smaller k, which thus increases the probability that the destination node successfully recovers n original packets in one segment.

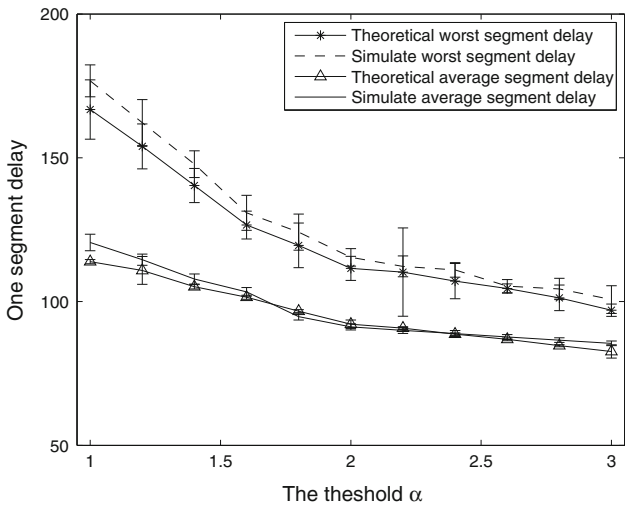

Fig. 4 The segment delay versus the threshold α for multicast

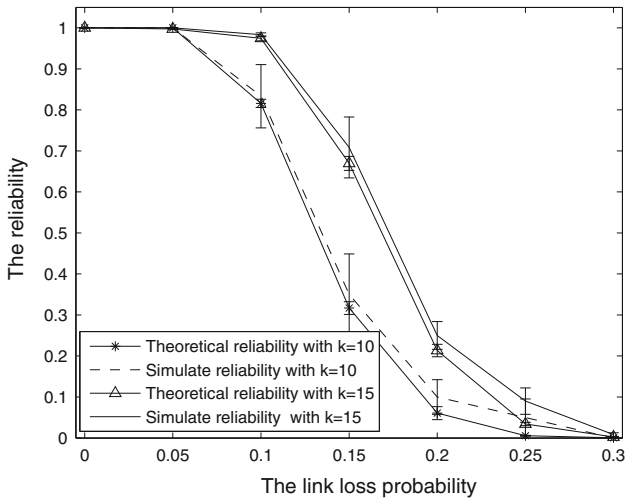

Fig. 5 The reliability versus the link loss probability q for unicast

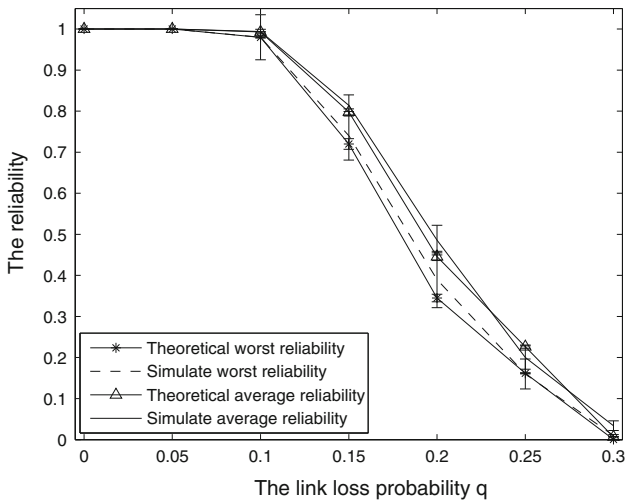

Fig. 6 The reliability versus the link loss probability q for multicast

In the second setting, we evaluate the reliability in multicast with $N = 10$. As shown in Fig. 6, both the worst reliability and the average reliability in our simulation are very close to that with theoretical results.

6.3 The impact of awake probability on segment delay

We now study the impact of sleep scheduling on the segment delay under different packet forwarding schemes. To fairly compare the performance, for non-coding schemes, e.g., optimal anycast scheme, anycast scheme and no-anycast-no-coding scheme, each packet in one segment will be sent $\lceil \frac{n+k}{n} \rceil$ times from the source s so as to guarantee at least $n + k$ transmissions for each segment. We vary the awake probability of network nodes in the range of [0.75, 1.0] for $n = 20, q = 0.1, t_b = 1, t_c = 2, \Delta t = 5, \alpha = 2.5$.

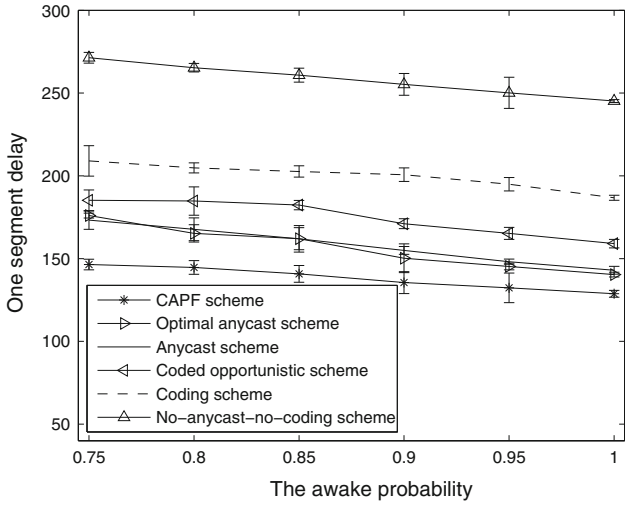

Fig. 7 The segment delay versus the awake probability p for unicast

Firstly, we evaluate the performance of the segment delay for unicast with $k = 30$. From Fig. 7, we can see that the segment delay with each of these schemes decreases with the increase of the awake probability, because the waiting delay at each hop decreases. We also can see that the segment delay with our CAPF scheme is the least among all the schemes.

Note that although with coded opportunistic scheme, its reliability for transmitting one packet to the destination might be better than other schemes, its segment delay does not perform the best. This is because to avoid collision, with coded opportunistic scheme, the current sending node cannot transmit the next packet until all the next hops finish their transmissions. Thus, more delay will be incurred by waiting the next hops' transmissions. Specifically, when the link loss probability is low, the segment delay with coded opportunistic scheme might be intolerable for delay-sensitive applications.

Secondly, we evaluate the performance of average segment delay and worst segment delay for multicast with $k = 50$ and $N = 10$. As shown in Fig. 8 where we omit the confidence intervals to clarify the figure, the segment delay with each of these schemes decreases with the increase of the awake probability, and the segment delay of our CAPF scheme is the smallest, evaluated with respect to both average segment delay and worst segment delay.

6.4 The tradeoff between delay and reliability

We now study the tradeoff between delay and reliability of our CAPF scheme with source coding. We vary the value of k in the range of $[5, 50]$ for $n = 20, p = 0.8, q = 0.1, t_b = 1, t_c = 2, \Delta t = 5, \alpha = 2.5$.

In the first setting, we evaluate the tradeoff in unicast. From Fig. 9, we can see that the reliability increases with the increase of k, because the number of coded packets

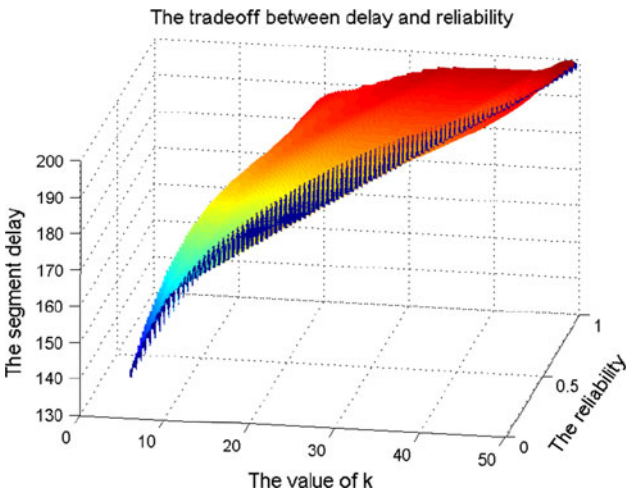

Fig. 9 The segment delay and reliability versus the value of k for unicast

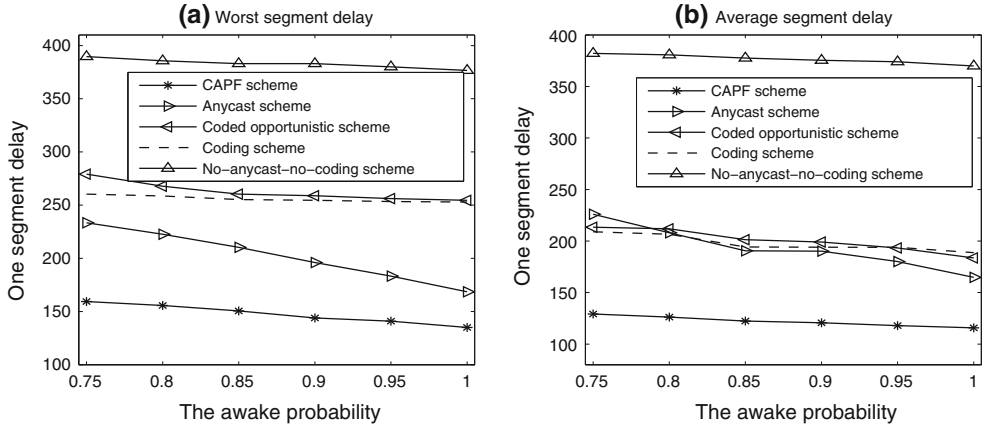

Fig. 8 The segment delay versus the awake probability p for multicast

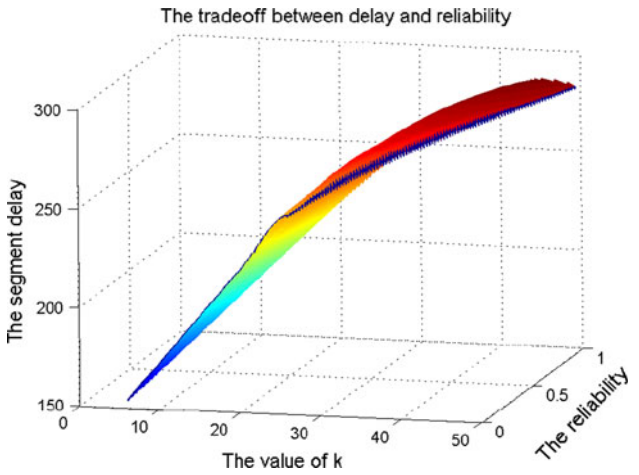

Fig. 10 The worst segment delay and reliability versus the value of k for multicast

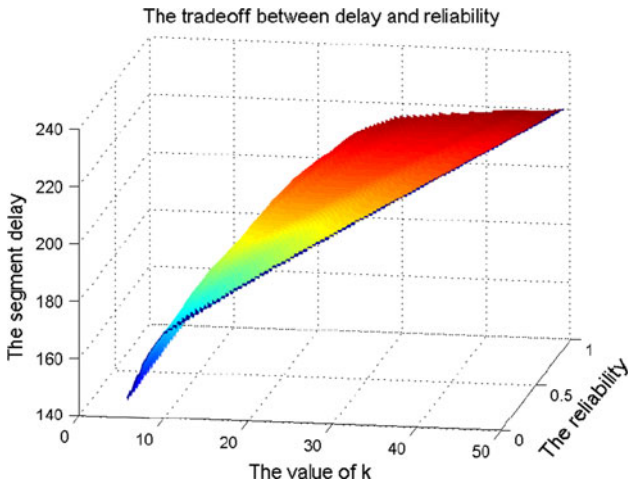

Fig. 11 The average segment delay and reliability versus the value of k for multicast

increases, which thus increases the probability that the destination node successfully recovers the n original packets of a segment. However, with the increase of k, the segment delay increases because it takes longer time for the source node to finish sending the packets of one segment.

We then evaluate the tradeoff between worst (average) segment delay and reliability for multicast with $N = 10$ in Fig. 10 (Fig. 11, respectively). We can observe the similar tradeoff as in the unicast case.

6.5 The comparison between source coding and network coding

We now study the segment delay and reliability in multicast with source coding and network coding for $N = 10$, $n = 10$, $k = 20$, $p = 0.8$, $t_b = 1$, $t_c = 2$, $\Delta t = 5$.

Firstly, we study the segment delay by varying the threshold α in the range of [1.0,3.0] with $q = 0.1$. Figures 12a, b give the worst and average segment delay of CAPF with coding over $GF(2^8)$ and $GF(2^5)$, respectively. To clarify the results, we omit the confidence intervals in this figure. We can see that when the Galois field is $GF(2^8)$, the segment delay with source coding is very close to that with network coding. Although network coding may increase the linear independence of source coding, the impact is negligible when the Galois field is $GF(2^8)$, because in this case the packets generated with source coding are linearly independent with a very high probability [21]. Thus, the packets with source coding and network coding are both linearly independent with the coded packets buffered at the destination(s) with a very high probability. In addition, the packet propagation delay with source coding is the same as that with network coding. So, when the Galois field is $GF(2^8)$, source coding and network coding have the similar segment delay. However, when the Galois field is $GF(2^5)$, network coding, to some extent,

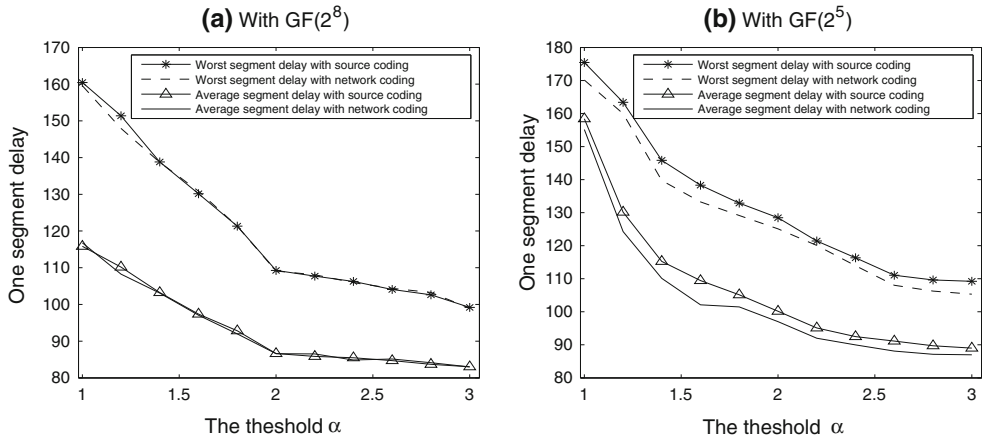

Fig. 12 The segment delay versus the threshold α for multicast

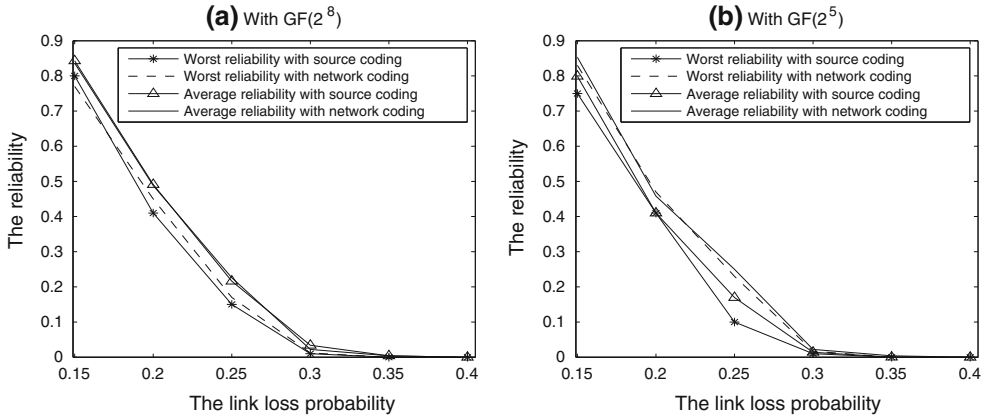

Fig. 13 The reliability versus the link loss probability q for multicast

increases the probability of the linear independence of packets generated with source coding, and thus decreases the segment delay.

In Fig. 13, we study the worst (average) reliability with both source coding and network coding by varying the link loss probability q in the range of [0.15, 0.4] with $\alpha = 1.5$. We can observe the similar results as in the segment delay.

7 Conclusions

In this paper, we propose a coded anycast packet forwarding (CAPF) scheme. To reduce the end-to-end delay, CAPF allows an intermediate node to maintain multiple next hops in its forwarding set. The sending node only needs to wait for any one of the next-hop nodes to wake up. Anycast alone, however, cannot support a good balance between end-to-end delay and reliability. To enhance the end-to-end reliability, CAPF introduces coding operations during packet propagation. To evaluate the performance of CAPF, we theoretically analyze the end-to-end delay, the reliability, and the tradeoff between them. Simulation results demonstrate that CAPF can provide a flexible mechanism to make good delay-reliability tradeoff and is effective to reduce the end-to-end delay and enhance the reliability.

Acknowledgments This paper was supported by the National Natural Science Foundation of China (NSFC) under Grant No. 61073038, and the National High-Tech Research and Development Plan of China under Grant No. 2009AA01A348.

References

1. Lu, G., Sadagopan, N., Krishnamachari, B., & Goel, A. (2005). Delay efficient sleep scheduling in wireless sensor networks. In *Proceedings of the 24th annual joint conference of the IEEE computer and communications societies, INFOCOM*, Miami, FL, USA, pp. 2470–2481.
2. Ye, W., Heidemann, J., & Estrin, D. (2002). An energy-efficient MAC protocol for wireless sensor networks. In: *Proceedings of the 21st annual joint conference of the IEEE computer and communications societies, INFOCOM*, Vol. 3, pp. 1567–1576.
3. Wireless mesh networking zigbee versus digimesh, available online: http://www.digi.com/pdf/wp_zigbeevsdigimesh.pd..
4. Liang, N., Chen, P., Sun, T., Yang, G., Chen, L., & Gerla, M. (2006). Impact of node heterogeneity in ZigBee mesh network routing. In: *IEEE international conference on systems, man and cybernetics, SMC '06, 1*, pp 187−191.
5. Kim, J., Lin, X., Shroff, N. B., & Sinha, P. (2010). Minimizing delay and maximizing lifetime for wireless sensor networks with anycast. *IEEE/ACM Transactions on Networking, 18*(2), 515–528.
6. Ahlswede, R., Cai, N., yen Robert Li, S., Yeung R. W., Member, S., & Member, S. (2000). Network information flow. *IEEE Transactions on Information Theory, 46*, 1204–1216.
7. Lun, D. S., Medard, M., & Koetter, R. (2006). Network coding for efficient wireless unicast. In: *Proceedings of the international Zurich seminar on communications*, pp. 74–77.
8. Zhan, C., Xu, Y., Wang, J., & Lee, V. (2009). Reliable multicast in wireless networks using network coding. In *Proceedings of the 6th international conference on mobile Adhoc and Sensor systems, MASS*, Macau, China, pp. 506–515.
9. Ghaderi, M., Towsley, D., & Kurose, J. (2008). Reliability gain of network coding in lossy wireless networks. In *Proceedings of the 27th IEEE conference on computer communications*, pp. 2171–2179.
10. Chachulski, S., Jennings, M., Katti, S., & Katabi, D. (2007). Trading structure for randomness in wireless opportunistic routing. In *Proceedings of the 2007 conference on applications, technologies, architectures, and protocols for computer communications, SIGCOMM*. New York, USA: ACM, pp. 169–180.
11. Biswas, S., & Morris, R. (2004). Opportunistic routing in multi-hop wireless networks. *SIGCOMM Computer Communication Review, 34*(1), 69–74.
12. Zorzi, M., & Rao, R. R. (2003). Geographic random forwarding (GeRaF) for ad hoc and sensor networks: Energy and latency performance. *IEEE Transactions on Mobile Computing, 2*(4), 349–365.
13. Wang, H. (2006). *Wireless sensor networks for acoustic monitoring*. Ph.D. dissertation, University of California at Los Angeles.
14. Lin, Y., Li, B., & Liang, B. (2008). Codeor: Opportunistic routing in wireless mesh networks with segmented network coding. In *Proceedings of the 16th IEEE international conference on network protocols*, Florida, pp. 13–22.
15. Liu, C., Wu, K., Xiao, Y., & Sun, B. (2006). Random coverage with guaranteed connectivity: Joint scheduling for wireless sensor

networks. *IEEE Transactions on Parallel and Distributed Systems, 17*(6):562–575.
16. Wicker, S. B. (2004). *Reed-solomon codes and their applications*. New York: IEEE Press.
17. Gallager, R. G. (1963). *Low-density parity-check codes*. Camberidge, MA: MIT Press.
18. Nguyen, U. T. (2008). On multicast routing in wireless mesh networks. *Computer Communications 31*(7):1385–1399.
19. Swapna, B. T., Eryilmaz, A., & Shroff, N. (June 2010). Throughput-delay analysis of random linear network coding for wireless broadcasting. In *Proceedings of the 2010 IEEE international symposium on network coding (NetCod 10)*, Toronton.
20. Ding, J., Sivalingam, K., Kashyapa, R., & Chuan, L. J. (2003). A multi-layered architecture and protocols for large-scale wireless sensor networks. In *Proceedings of IEEE 58th vehicular technology conference*, October 6–9, Vol. 3, pp 1443–1447.
21. Ma, G., Xu, Y., Lin, M., & Xuan, Y. (2007, January). A Content Distribution system based on sparse linear network coding. In *Proceedings of the third workshop on network coding (Netcod)*.

Author Biographies

Xiumin Wang received her B.S. from the Department of Computer Science, Anhui Normal University, China, in 2006. She is currently working toward the Ph.D. degree at the School of Computer Science and Technology, University of Science and Technology of China, Hefei, China. She is also under a joint Ph.D. program offered by University of Science and Technology of China and City University of Hong Kong. Her research interests include wireless networks, routing design, and network coding.

Kui Wu received the B.Sc. and the M.Sc. degrees in Computer Science from Wuhan University, China in 1990 and 1993, respectively, and the Ph.D. degree in Computing Science from the University of Alberta, Canada, in 2002. He joined the Department of Computer Science at the University of Victoria, Canada in the same year and is currently an Associate Professor there. His research interests include mobile and wireless networks, network performance evaluation, and network security.

Jianping Wang received the B.Sc. and the M.Sc. degrees in computer science from Nankai University, Tianjin, China in 1996 and 1999, respectively, and the Ph.D. degree in computer science from the University of Texas at Dallas in 2003. She is currently an assistant professor at the Department of Computer Science, City University of Hong Kong. Jianping's research interests include optical networks and wireless networks.

Yinlong Xu received his B.S. in Mathematics from Peking University in 1983, and MS and Ph.D. in Computer Science from university of Science and Technology of China (USTC) in 1989 and 2004 respectively. He is currently a professor with the School of Computer Science and Technology at USTC. Prior to that, he served the Department of Computer Science and Technology at USTC as an assistant professor, a lecturer, and an associate professor. Currently, he is leading a group of research students in doing some networking and high performance computing research. His research interests include network coding, wireless network, combinatorial optimization, design and analysis of parallel algorithm, parallel programming tools, etc. He received the Excellent Ph.D. Advisor Award of Chinese Academy of Sciences in 2006.