Joint Congestion Control and Scheduling in Wireless Networks With Network Coding

Ronghui Hou, King-Shan Lui, *Senior Member, IEEE*, and Jiandong Li, *Senior Member, IEEE*

***Abstract*—This paper studies how to perform joint congestion control and scheduling with network coding in wireless networks. Under network coding, a node may need to buffer a sent packet for decoding a packet to be received later. If sent packets are not forgotten smartly, much buffer space will be taken up, leading to dropping of new incoming packets. This unexpected packet dropping harms the final throughput obtained, although optimal scheduling has been used. To solve the problem, we introduce a new node model incorporating a transmission-mode preassignment procedure and a scheduling procedure. The introduced transmission-mode preassignment avoids memorizing several sent packets to reduce buffer overhead. We develop a new scheduling policy based on our node model and analyze formally the stability property of a network system using the proposed policy. We finally evaluate the efficiency of our algorithm through simulations from the perspectives of throughput and packet loss ratio.**

***Index Terms*—Congestion control, cross-layer design, network coding, scheduling, wireless interference.**

I. Introduction

RECENTLY, new techniques have been proposed to utilize intelligently wireless interference to improve network throughput. Network coding exploits the wireless medium broadcast nature to improve network capacity. Generally, network coding can be classified into two different categories: *intrasession* and *intersession*. Intrasession network coding is performed on packets from the same session, whereas intersession network coding is performed on packets across different sessions. This paper considers the COPE-style intersession network coding. Consider a set of receivers that each wants to receive its desired packet from a common sender. If each receiver has all the other packets except the desired one, the common sender can code all the packets and transmit a single coded packet to all receivers, and each receiver can successfully decode the desired one. Following [1] and [2], we consider network coding without opportunistic listening, which allows network coding between two flows only. Fig. 1, which is adapted from [3, Fig. 1], shows the coding opportunity at Node 2. There are two flows going on $\langle 1, 2, 3\rangle$ and $\langle 3, 2, 1\rangle$, respectively. After Node 2 receives packets p_1 and p_2 from both 1 and 3, respectively, Node 2 can code the packets and transmit the coded packet to both 1 and 3. Since Node 1 has sent p_1 before, Node 1 can decode p_2 from the coded packet $p_1 \oplus p_2$. Similarly, Node 3 can decode p_1.

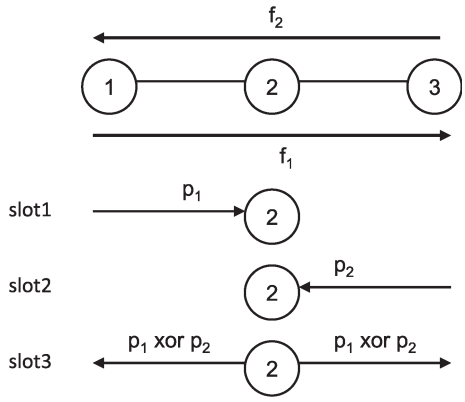


Fig. 1. Network coding.

It is well known that an intelligent scheduling policy is very important to improve network throughput. Scheduling in wireless networks is a big challenge due to wireless interference. Many works study scheduling issues in wireless networks [4]. Network coding further complicates the issue. In addition to determining which link should be active, each node also needs to determine which transmission mode (traditional or coded) should be used at the current time. For instance, assume that the data rates of f_1 and f_2 in Fig. 1 are $(1/4)$ and $(3/8)$, respectively. The link capacity is one unit. In this scenario, if Node 2 wants to code all the packets of f_2, packets will be backlogged. The queue becomes overflowed, whereas the link capacities are not fully utilized. A better way is to code f_1 and f_2 at a rate of $(1/4)$ and send the remaining packets of f_2 in a traditional manner. On the other hand, under the stochastic traffic model, it is possible that Node 2 gets several packets of f_1 before getting one packet of f_2. Node 2 should determine whether transmitting a packet of f_1 traditionally or waiting for the future coding opportunity. An efficient scheduling policy should utilize all the possible coding opportunities to improve network throughput.

The scheduling policy proposed in [5] uses the queue length at each node to determine which link should be active at the current time slot. The problem of this scheme is that each node needs to keep each packet sent by itself earlier until the next

Manuscript received July 4, 2013; revised November 12, 2013; accepted December 23, 2013. Date of publication February 10, 2014; date of current version September 11, 2014. This work was supported in part by the National Natural Science Foundation of China under Grant 61231008 and Grant 61101143, by the Important National Science and Technology Projects under Grant 2013ZX03004007-003, by the Hong Kong University Small Project Fund Under Grant 104001905, by the Fundamental Research Funds for the Central Universities under Grant K50511010006, and by the 111 Project under Grant B08038. The review of this paper was coordinated by Prof. N. Kato.

R. Hou and J. Li are with the State Key Laboratory of Integrated Service Networks, Xidian University, Xi'an 710071, China.

K.-S. Lui is with the Department of Electrical and Electronic Engineering, The University of Hong Kong, Kowloon, Hong Kong.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVT.2014.2298404

hop transmits the packet. This is because a node does not know whether the packet it sent previously will be used for coding by the next hop node until the next hop node sends it out. For instance, in Fig. 1, after Node 1 sends p_1, p_1 should be kept at Node 1 until Node 2 transmits p_1. Since Node 2 transmits p_2 using network coding, Node 1 uses p_1 to decode p_2. When the average data rates of the flows differ a lot, many packets of a flow would be transmitted in the traditional manner. Although these packets can be forgotten right after they are sent because the sending node does not know it until these packets are sent by the next hop, the sending node has to buffer all of them for a while. The space overhead is thus increased.

In practice, some nodes may have limited storage resources. For instance, in multihop cellular networks, each node is a smartphone or tablet, and the storage resources are limited [6], [7]. The future incoming packets would be dropped when the node space allocated for packet forwarding is full, which severely affects network throughput performance. We would like to use an example to illustrate this phenomenon. Suppose that the buffer size is 50 packets, which is the default value used in ns2 [8]. Then, consider the network state that Node 1 has 30 packets to be transmitted and needs to keep another 20 sent packets used for future decoding. Since there is no more room in the buffer, Node 1 would drop new packets coming from the upper layer due to overflow in default [7]. However, it is possible that only a few out of the 20 sent packets would be transmitted by Node 2 in a coded manner. If Node 1 knows which packet would be transmitted traditionally in advance, the node can forget the packet immediately to accept more new packets from the upper layer. If the packet loss happens in the intermediate forwarding node, the bandwidth resources consumed for the packet transmission over the previous hops would be wasted. This hinders network throughput. With the declining of the memory chip prices, we may say that routing buffers can be overprovisioned. However, large routing buffer increases queuing delay [8], [9]. Therefore, it is desirable to forget a sent packet as soon as possible so that a large buffer is not needed.

Motivated by this issue, this paper aims at reducing the space overhead while providing high network throughput. We propose the transmission-mode preassignment procedure that tells the node which transmission mode (network coding or traditionally) the next hop will use to transmit a packet immediately after the packet arrives the next hop. The next hop can then inform the previous hop to forget packets that are going to be transmitted traditionally. This information can be carried by the signaling messages. Signaling message delivery schemes used by existing protocol [3] can be used in our mechanism.

To develop a joint scheduling-and-transmission-mode assignment scheme, we introduce a new node model so that we can formulate the problem as a network utility maximization problem. We then apply the "layering as optimization decomposition" technique [10] to decompose the linear programming optimization into several subproblems, and each layer corresponds to a decomposable subproblem. The theoretical decomposition motivates us to develop a practical cross-layer optimization algorithm. In our algorithm, each source determines the instantaneous data rate injected into the network based on the current network state; then, a feasible set of links is selected to transmit at the current time. Both procedures adjust to each other to optimize the network utility function.

II. Related Works

With the rising demand of bandwidth under limited available spectrum, using an intelligent resource-allocation scheme in wireless networks has received substantial attention. Scheduling, which selects a set of links to be active without conflict, is an important issue in allocating the bandwidth resources. The scheduling problem is in general NP-hard [4], and several suboptimal scheduling solutions are proposed. There have been some works studying the cross-layer issue on congestion and scheduling in wireless networks. The work in [11] analyzes the effect of imperfect scheduling on congestion control in multihop wireless networks, and it shows that the cross-layer approach outperforms the layered approach. In [12], the joint approach of queue-length-based scheduling and congestion control in cellular networks is proposed, where the channel may vary according to time or according to receivers. In [13], a distributed scheduling algorithm and a congestion control mechanism to achieve the approximate optimal solution are described. Unfortunately, the authors consider the specific one-hop interference model, which may not be applicable in a general network. In [14], the joint problem of multipath routing and link-level reliability in multihop wireless networks is studied. The authors develop a decentralized scheduling policy that selects an appropriate channel code rate for each link to cope with different degrees of data reliability among the different links. In [15]–[19], the control in multichannel multiradio wireless networks is studied. In [15] and [16], mixed-integer linear programming models for the joint of congestion control, routing, and scheduling in multichannel wireless networks are presented. However, solving the linear programming is NP-hard, and no algorithm was proposed. The work in [17] proposes a distributed scheduling algorithm in multichannel wireless networks. In [18], a cross-layer optimization algorithm, which relies on the scheduling algorithm in [17] as lower layer solutions, is proposed. In [19], the same problem as [18] is considered, and it presents a new tuple-based network model to facilitate decoupling the optimization on different layers. None of the given works consider network coding.

Some works consider the scheduling problem with network coding with various objectives in relay-based cellular networks. In relay-based cellular networks, a relay station forwards downlink traffic to user and uplink traffic to a base station. Thus, many coding opportunities may exist at the relay station. As the transmission rate is bounded by the minimum rate among receivers, coding too many native packets together may not be beneficial. In [20], this tradeoff is considered, and a scheduling algorithm to specify which packets should be coded by the relay station is proposed. In [21], the tradeoff between reducing packet delay and saving energy is considered. To reduce energy consumption, a node should code as much as possible. The transmission of a packet may have to be delayed to wait for a coding opportunity. Therefore, in [21], an opportunistic scheduling to determine whether the relay station should

transmit (coded or noncoded) packet or wait at a certain time slot is proposed. In [22], it is considered that the relay station applies network coding to reduce the number of retransmissions and proposes a scheduling algorithm to determine which packets the relay station should code. In [23], the power control, channel allocation, and link scheduling problems are jointly considered, and an opportunistic resource scheduling algorithm is proposed. In [24], the scheduling problem for broadcast traffic with network coding in relay-based networks is studied. We can see that different works concern different optimization objectives. In [23], the same objective as this paper is considered. However, in [23], the buffer overhead at a user node is not considered, which is particularly important for cellular networks.

Many works study scheduling with network coding in multihop wireless networks. In [2], [25], and [26], the theoretical throughput gain obtained by network coding in wireless networks is analyzed, and in [27], the joint congestion control, routing, and scheduling is formulated as a linear programming problem, whereas no algorithm was proposed to solve the problem. In [5], a scheduling policy with network coding is proposed. In addition to network coding, the scheduling policy proposed in [28] also considers the energy consumption on each link, the packet loss probability, and the transmission rate on each link. If we do not consider energy consumption and packet loss probability, the proposed scheduling scheme is reduced to the algorithm in [5]. In [2], k-tuple coding is introduced, and 2-tuple coding is the same as COPE-style coding. The scheduling policy used in [2] is the same as that in [5] when considering 2-tuple coding. In [1], *pairwise intersession network coding* is considered, which is different from the COPE-style network coding model. In [29], the tradeoff between delay and throughput with network coding in multihop wireless networks is studied. Nevertheless, the buffer issue was not mentioned in the above works.

III. Model and Assumptions

We consider a multihop wireless network with the set of nodes $\mathcal{N}$. Let $\mathcal{L}$ be the set of links, and each link $l = (i, j) \in \mathcal{L}$ denotes that node j can successfully receive the data from node i when there is no interference. Let $\mathcal{F}$ denote the set of traffic flows. Following [11], [30], and [31], each flow is served by a single path, and the path is predetermined. Let $s(f)$ and $d(f)$ be the source and the destination of the flow f, respectively. If f goes through intermediate node i, denote $p^f(i)$ and $s^f(i)$ as the predecessor and successor of i on the path carrying f, respectively. Based on the routes of the traffic flows, we can determine all the coding opportunities at each node. For example, given a node i carrying two flows f_1 and f_2, if $p^{f_1}(i) = s^{f_2}(i)$ and $p^{f_2}(i) = s^{f_1}(i)$, node i can code the packets of f_1 and f_2, respectively. It is possible that node i can also code f_1 with another flow f_3, apart from f_2. To clearly define each coding opportunity, we introduce the definition of *coding structure*. If node i can code flows f_1 and f_2 with the next hops u and v, respectively, we have a coding structure $\phi = \{i, [u; v], [f_1; f_2]\}$ at node i. ϕ defines a pair of native links, i.e., $\{(i, u), (i, v)\}$, called *coding link*. It is possible that

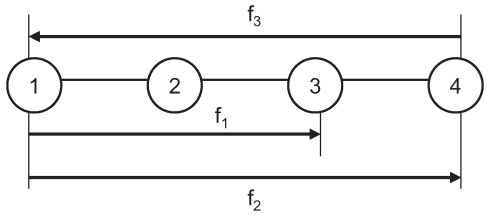


Fig. 2. Coding structure.

there are multiple coding structures at node i. For instance, in Fig. 2, there are two coding structures at node 2, $\phi_1 = \{2, [1; 3], [f_3; f_1]\}$ and $\phi_2 = \{2, [1; 3], [f_3; f_2]\}$. ϕ_1 tells that Node 2 can code f_3 and f_1 and that the next hops of f_3 and f_1 are Nodes 1 and 3, respectively. ϕ_1 and ϕ_2 specify the same next hop information but different traffic flows. Let $\Phi(i)$ be the set of coding structures at node i. We write $f \in \phi$ if it is one of the two flows in coding structure ϕ.

We assume that time is divided into slots. Following [5], let $\mathcal{M}$ denote the set of feasible scheduling decisions or schedules at a certain time slot. Each element of $\mathcal{M}$ defines 1) the links or coding structures that are active in the time slot, 2) the flows that are carried on the active links and coding structures, and 3) the rates of the flows being carried. For instance, Fig. 1 shows the feasible schedules of three time slots. Link (1, 2) carries f_1 in Slot 1, link (3, 2) carries f_2 in Slot 2, and coding structure $\{2, [1; 3], [f_2, f_1]\}$ is active in Slot 3.

We denote by $r_l^f(M)$ the rate of native link l serving f under a feasible schedule M. Denote by $r_{i,\phi}(M)$ the rate of coding structure ϕ at node i under schedule M. The link rate depends on the specific interference model [32]. In the protocol interference model, a transmission on link (i, j) at the negotiated rate is successful if none of the nodes in the interference range of j is transmitting. Thus, $r_l^f(M)$ is either a fixed rate or zero under the protocol interference model. If we consider the physical layer interference model, the data link rate depends on the signal-to-interference-plus-noise ratio at the receiver, which is often approximated by the Shannon formula. The rate of a link may be different under the different schedule. We denote by $\mathcal{R}$ the feasible rate region. Each element in $\mathcal{R}$ is a rate vector $\vec{r}(M)$, containing all $r^f_{(i,\, s^f(i))}(M)$ and $r_{i,\phi}(M)$, which is yielded by a feasible schedule M. In other words, $\mathcal{R}$ includes the rate vectors produced by all the possible feasible schedules in $\mathcal{M}$. To remove context ambiguity, we use $\vec{r} \in \mathcal{R}$ to denote a feasible rate vector, which is determined by a feasible schedule.

Let $\lambda(f)$ be the input rate of the flow $f \in \mathcal{F}$, which falls in the region of $(0, \Lambda_f]$, and we assume that Λ_f is known in advance [11], [18]. Define the utility function for flow f as $U_f(\lambda_f)$, which reflects the level of "satisfaction" of flow f when its data rate is λ_f. Following [11], $U_f(\cdot)$ is assumed to be strictly concave, nondecreasing, and twice continuously differentiable on $(0, \Lambda_f]$. Our algorithm aims at maximizing the network utility by jointly considering congestion control, transmission-mode assignment, and scheduling.

IV. Cross-Layer Optimization

Here, we first describe the existing scheduling scheme in wireless networks with network coding [2], [5]. Since the

existing scheduling scheme does not consider the buffer issue, we introduce transmission-mode assignment procedure and propose a new node model. Afterward, the problem formulation is presented. We then develop a suboptimal cross-layer optimization algorithm. Finally, we theoretically analyze the input rate region supported by the proposed scheduling policy.

A. Existing Scheduling With Network Coding

A scheduling policy is defined as an algorithm that chooses a feasible schedule $M \in \mathcal{M}$ in each time slot t [5]. We denote by $q_i^f(t)$ the number of packets of flow f in the queue of node i at the beginning of time slot t. The scheduling policy in [5] is as follows:

$$M^*(t) = \arg\max_{M\in\mathcal{M}} \left\{ \sum_i \sum_f \left(q_i^f(t) - q_{s^f(i)}^f(t) \right) r_{(i,s^f(i))}^f(M) \right\} \quad (1)$$

The stability region Λ of the network is the set of all arrival rate vectors $\vec{\lambda}$ that can be supported while ensuring that all packet queues in the network remain finite. In [2] and [5], it is shown that the scheduling policy shown by (1) stabilizes the network for all arrival rate vectors inside Λ. Denote by $\mathcal{M}_{\text{code}}$ the set of feasible schedules that considers network coding and $\mathcal{M}_{\text{no\_code}}$ by that without network coding. We can say that $\mathcal{M}_{\text{no\_code}}$ is a subset of $\mathcal{M}_{\text{code}}$, and some schedules in $\mathcal{M}_{\text{code}}$ may not be feasible when not considering network coding (see Fig. 1 for any two links that interfere with each other). A feasible schedule in $\mathcal{M}_{\text{code}}$ can contain two links (2, 1) and (2, 3), whereas any feasible schedule in $\mathcal{M}_{\text{no\_code}}$ cannot.

By using (1), a node cannot know in advance which transmission mode the next hop will use to transmit a packet sent by itself. Considering the stochastic traffic model, it is possible that Node 2 has received several packets of f_1 but no packet of f_2 in Fig. 1. Let us assume that $q_2^{f_1}(t) > q_1^{f_1}(t) > q_3^{f_2}(t)$. When $q_2^{f_2}(t) = 0$, according to (1), $M^* = \{(2, 3)\}$ since $q_2^{f_1}(t)$ is the largest. Node 2 will transmit traditionally the Head of Line (HoL) packet of f_1 at time t. We assume that the transmission rate of each link in Fig. 1 is the same. When $q_2^{f_2}(t) > 0$, node 2 can code the packets from f_1 and f_2; thus, $q_2^{f_1}(t) r_{(2,3)}^f + q_2^{f_2}(t) r_{(2,1)}^f$ is the largest. In this case, the HoL packet of f_1 would be transmitted being coded with another packet of f_2. That is to say, a node does not know which transmission model would be assigned for the buffered packets until they are scheduled. Thus, the node should keep each packet until the next hop completes transmitting the packet, to assure that each coded packet would be decoded. In particular, when the average data rates of two flows differ a lot, many packets of a flow would be probably transmitted traditionally, such that a large unnecessary buffer is required by network coding. This is undesirable because the node will drop some new data packets due to buffer overflow. Consequently, we propose the transmission-mode preassignment procedure.

B. Node Model

To reduce the buffer size, this paper introduces the procedure of *transmission-mode preassignment*. After a node receives a packet, it decides immediately whether the packet should be transmitted in a coded manner or traditionally. If the packet would be transmitted traditionally, the previous hop does not have to memorize the packet then. To achieve the above practice, we propose a node model. A node would put packets of different statuses in different queues, indicating they are going to be handled in different ways. That is, for each flow f that goes through node i, i maintains the following.

1) *Input queue for keeping incoming packets (denoted X_i^f).* Packets in this queue have yet to be assigned a transmission mode. Packets here will be moved to one of the output queues after the transmission mode is determined.
2) *Output queue for packets to be sent out traditionally (denoted W_i^f).* Packets in this queue have been determined to be transmitted in a traditionally manner and are waiting for its turn to be sent out to the channel. i can inform its neighbors to forget these packets.
3) *Output queue for packets of each coding structure ϕ where $\phi \in \Phi(i)$ (denoted $Z_{i,\phi}$).* The packets of flow $f \in \phi$ moved to this queue are to be transmitted in coded manner. When Node i move a packet of f to $Z_{i,\phi}$, a new code packet will be formed immediately if there is a native packet of another flow $f' \in \phi$ in $Z_{i,\phi}$. In other words, $Z_{i,\phi}$ contains two kinds of packets. One is the coded packet, and another one is the native packet of a flow waiting to be coded together. All the packets are waiting for their turn to be sent out to the channel.

Both output queues push traffic to physical link $(i, s^f(i))$. The transmission-mode assignment involves deciding to which output queue (W_i^f or $Z_{i,\phi}$, $f \in \phi$) a packet from the input queue (X_i^f) should be put. A wise decision should not overwhelm any output queue while maximizing the utility. To model the relation between the queues, we represent each queue as a virtual node. We put a virtual link from one queue to another one if packets can be moved in that manner. Each virtual link is associated with a rate that tells the average ratio of the packets being moved to each output queue.

Fig. 3 shows the model of Node 2 in Fig. 2. In the figure, a square box is a queue, and a virtual link is represented as a dashed arrow. Node 2 forwards three flows; therefore, it has three input queues. The coding structure ϕ_1 contains f_1 and f_3; therefore, there are two virtual links from X_2^1 to Z_{2,ϕ_1} and from X_2^3 to Z_{2,ϕ_1}, respectively. Similarly, ϕ_2 contains f_2 and f_3, and there are two virtual links from X_2^2 to Z_{2,ϕ_2} and from X_2^3 to Z_{2,ϕ_2}, respectively. It is obvious that the optimal transmission-mode assignment should move the same number of packets of f_1 and f_3 to Z_{2,ϕ_1}.

Note that all the packets of the source node of f are transmitted traditionally; therefore, the proposed node model is not applied at $s(f)$. In other words, there is one output queue of f at $s(f)$. When the upper layer injected new packets, all the packets are immediately moved to the output queue $W_{s(f)}^f$.

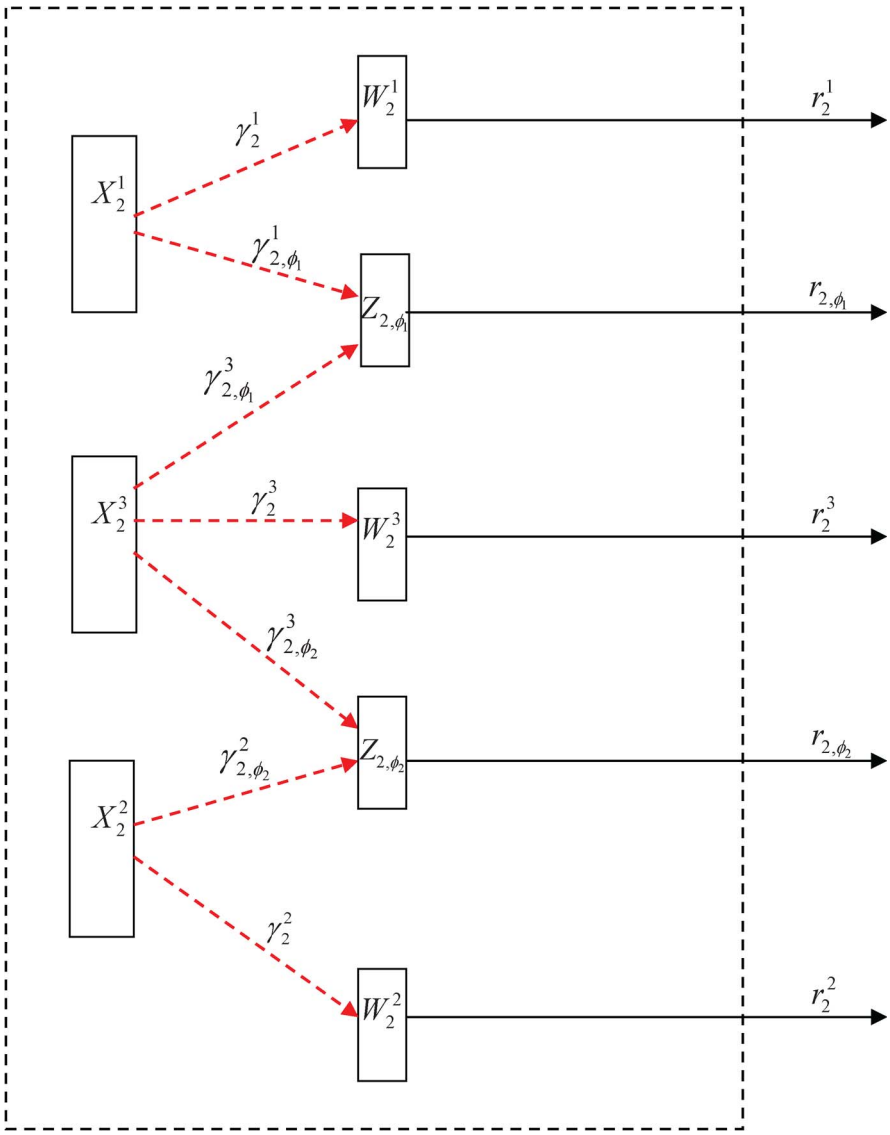


Fig. 3. Node model.

Let $\gamma_{i,\phi}^f$ denote the rate on the virtual link from X_i^f to $Z_{i,\phi}$. For $f \in \phi$ and $f' \in \phi$, we should have $\gamma_{i,\phi}^f = \gamma_{i,\phi}^{f'}$ in the optimal solution. Let W_i^f denote the queue containing the packets to be transmitted traditionally. The rate of the virtual link from X_i^f to W_i^f is γ_i^f. The rate of the traffic leaving input queue X_i^f is thus $\sum_{\phi \in \Phi(i),\, f \in \phi} \gamma_{i,\phi}^f + \gamma_i^f$. To assure the stability of the system, $\vec{\gamma}$ is bounded by the rate on the physical links. For instance, $\gamma_{i,\phi}^f$ should not be larger than $r_{i,\phi}$; otherwise, $Z_{i,\phi}$ would be infinitely large. Denote by Γ the feasible rate region for $\vec{\gamma}$, which will be defined in Section VI, based on the throughput-optimal and stability arguments.

C. Problem Formulation

Based on the proposed node model, the problem concerned in this paper is formulated as

$$\text{maximize} \quad \sum_{f \in \mathcal{F}} U_f(\lambda_f) \tag{2}$$

$$\text{subject to} \quad \sum_{\phi \in \Phi(p^f(i)), f \in \phi} r_{p^f(i),\phi} + r_{p^f(i)}^f \leq \sum_{\phi \in \Phi(i),\, f \in \phi} \gamma_{i,\phi}^f + \gamma_i^f \;\forall\, i \neq s(f), \forall\, f \tag{3}$$

$$\gamma_{i,\phi}^f \leq r_{i,\phi} \;\forall\, i,\; \forall\, \phi \in \Phi(i), \forall\, f \in \phi \tag{4}$$

$$\gamma_i^f \leq r_i^f \;\forall\, i \neq s(f),\; \forall\, f \tag{5}$$

$$\lambda_f \leq r_{s(f)}^f \;\forall\, f \tag{6}$$

$$\vec{r} \in \mathcal{R} \tag{7}$$

$$\vec{\gamma} \in \Gamma. \tag{8}$$

Our objective is to maximize the sum of the utility functions of the data rates for all flows. Equation (3) makes sure the rate of incoming traffic to queue X_i^f is no larger than the rate that traffic leaves the queue; otherwise, the input queue would go to infinity. Similarly, (4) ensures that the rate of incoming packets of f to queue $Z_{i,\phi}$ does not exceed the rate on coding structure ϕ. Equation (5) assures that the input rate to W_i^f is no larger

than its output rate. In other words, (3)–(5) guarantee that the queues of each node would not be infinite.

$$\begin{aligned} L(\vec{\lambda},\vec{\gamma},\vec{r},\vec{\beta}) = &\sum_f U_f(\lambda_f) + \sum_{f,\, i\neq s(f)} \beta^f_{i,\,\mathrm{in}} \\ &\times \left(\sum_{\phi\in\Phi(i),\, f\in\phi} \gamma^f_{i,\,\phi} + \gamma^f_i \right. \\ &\qquad \left. - \sum_{\phi\in\Phi(p^f(i)),\, f\in\phi} r_{p^f(i),\,\phi} - r^f_{p^f(i)} \right) \\ &+ \sum_{i,\,\phi\in\Phi(i),\, f\in\phi} \beta^f_{i,\,\phi,\,\mathrm{out}} \left(r_{i,\,\phi} - \gamma^f_{i,\,\phi} \right) \\ &+ \sum_{f,\, i\neq s(f)} \beta^f_{i,\,\mathrm{out}} \left(r^f_i - \gamma^f_i \right) \\ &+ \sum_f \beta^f_{s(f),\,\mathrm{out}} \left(r^f_{s(f)} - \lambda_f \right) \\ &\vec{r}\in\mathcal{R} \\ &\vec{\gamma}\in\Gamma. \end{aligned} \tag{9}$$

The optimal solution for (2) tells the maximum data rate of each flow and the data rate on each link or each coding structure, We would like to use a Lagrangian dual decomposition method to solve our problem, Corresponding to constraints (3)–(6), we define Lagrange multipliers $\beta^f_{i,\,\mathrm{in}}$, $\beta^f_{i,\,\phi,\,\mathrm{out}}$, $\beta^f_{i,\,\mathrm{out}}$, and $\beta^f_{s(f),\,\mathrm{out}}$, respectively [33]. Later, we will see that these multipliers reflect the queue sizes. The Lagrangian is (9), and the dual of problem (2) is

$$\min D(\vec{\beta}) \tag{10}$$

, where $G_f(\vec{\beta})$ is a function of λ_f, $V_1(\vec{\beta})$ is a function of γ_i and $\gamma^f_{i,\,\phi}$, and $V_2(\vec{\beta})$ is a function r^f_i and $r_{i,\,\phi}$, and

$$\begin{aligned} D(\vec{\beta}) &= \max_{\vec{\lambda},\vec{\gamma},\vec{r}} L(\vec{\lambda},\vec{\gamma},\vec{r},\vec{\beta}) \\ &= \sum_{f\in\mathcal{F}} \max_{0<\lambda_f\le\Lambda_f} G_f(\vec{\beta}) \\ &\quad + \max_{\vec{\gamma}\in\Gamma} V_1(\vec{\beta}) + \max_{\vec{r}\in\mathcal{R}} V_2(\vec{\beta}) \\ G_f(\vec{\beta}) &= U_f(\lambda_f) - \beta^f_{s(f),\,\mathrm{out}}\lambda_f \\ V_1(\vec{\beta}) &= \sum_{i\in\mathcal{N},\phi\in\Phi(i),f\in\phi} \left(\beta^f_{i,\,\mathrm{in}} - \beta^f_{i,\,\phi,\,\mathrm{out}}\right)\gamma^f_{i,\,\phi} \\ &\quad + \sum_{f\in\mathcal{F},i\in\mathcal{N},i\neq s(f)} \left(\beta^f_{i,\,\mathrm{in}} - \beta^f_{i,\,\mathrm{out}}\right)\gamma^f_i \\ V_2(\vec{\beta}) &= \sum_{i,\phi\in\Phi(i)} \left(\sum_{f\in\phi} \left(\beta^f_{i,\,\phi,\,\mathrm{out}} - \beta^f_{s^f(i),\mathrm{in}}\right) \right) r_{i,\,\phi} \\ &\quad + \sum_{i,f} \left(\beta^f_{i,\,\mathrm{out}} - \beta^f_{s^f(i),\mathrm{in}}\right) r^f_i. \end{aligned} \tag{11}$$

Since the dual objective function $D(\vec{\beta})$ is convex, we use the subgradient method to solve the dual problem [11], [33]. Generally, the subgradient method uses the iteration

$$\vec{\beta}(t+1) = \vec{\beta}(t) - \frac{D(\vec{\beta})(t)}{\partial\vec{\beta}} \tag{12}$$

where t is the iteration index. We thus derive the updating rules for the Lagrange multipliers as follows:

$$\begin{aligned} &\beta^f_{i,\,\mathrm{in}}(t+1) \\ &= \left[\beta^f_{i,\,\mathrm{in}}(t) + \alpha^f_i \left(\sum_{\phi\in\Phi(p^f(i)),\, f\in\phi} r_{p^f(i),\,\phi}(t) + r^f_{p^f(i)}(t) \right.\right. \\ &\qquad \left.\left. - \sum_{\phi\in\Phi(i)} \gamma^f_{i,\,\phi}(t) - \gamma^f_i(t) \right) \right]^+ \end{aligned} \tag{13}$$

$$\begin{aligned} &\beta^f_{i,\,\phi,\,\mathrm{out}}(t+1) \\ &= \left[\beta^f_{i,\,\phi,\,\mathrm{out}}(t) + \alpha^f_{i,\,\phi,\,\mathrm{out}} \left(\gamma^f_{i,\,\phi}(t) - r_{i,\,\phi}(t) \right) \right]^+ \end{aligned} \tag{14}$$

$$\begin{aligned} &\beta^f_{i,\,\mathrm{out}}(t+1) \\ &= \left[\beta^f_{i,\,\mathrm{out}}(t) + \alpha^f_{i,\,\mathrm{out}} \left(\gamma^f_i(t) - r^f_i(t) \right) \right]^+ \end{aligned} \tag{15}$$

$$\begin{aligned} &\beta^f_{s(f),\,\mathrm{out}}(t+1) \\ &= \left[\beta^f_{s(f),\,\mathrm{out}}(t) + \alpha^f_{i,\,\mathrm{out}} \left(\lambda_f(t) - r^f_{s(f)}(t) \right) \right]^+. \end{aligned} \tag{16}$$

Note that the update method for $\beta^f_{s(f),\,\mathrm{out}}(t)$ is different from $\beta^f_{i,\,\mathrm{out}}(t)$, $i \neq s(f)$, as shown in (15) and (16). Equations (17)–(19) tell how to find $\vec{\lambda}$, $\vec{r}$, and $\vec{\gamma}$ at different t required, as shown in the following:

$$\lambda_f(t) = \arg \max_{0<\lambda_f\le\Lambda_f} U_f(\lambda_f) - \beta^f_{s(f),\,\mathrm{out}}(t)\lambda_f \tag{17}$$

$$\vec{\gamma}(t) = \arg\max_{\vec{\gamma}\in\Gamma} V_1\left(\vec{\beta}(t)\right) \tag{18}$$

$$\vec{r}(t) = \arg\max_{\vec{r}\in\mathcal{R}} V_2\left(\vec{\beta}(t)\right). \tag{19}$$

α^f_i, $\alpha^f_{i,\,\phi,\,\mathrm{out}}$, and $\alpha^f_{i,\,\mathrm{out}}$ are iterative step sizes. $[\bullet]^+ = \max\{\bullet, 0\}$. That is, all Lagrangian multipliers must not be negative. The process continues until $\vec{\beta}$ converges. Assume that $\mathcal{R}$ and Γ are convex. When the step size is set appropriately small, $\vec{\lambda}$, $\vec{\gamma}$, and $\vec{r}$ converge to the optimal solution [11].

The optimal solution requires the feasible rate region $\mathcal{R}$ to be predetermined, which is NP-complete. Even if we have $\mathcal{R}$, calculating $\vec{\gamma}(t)$ and $\vec{r}(t)$ is still difficult. Moreover, the computational overhead would be huge due to a small iterative step size. Nevertheless, the subgradient method for the Lagrangian dual problem has an attractive decomposition property: For each t, (17) determines the data rate of each flow (congestion control component), (18) assigns the transmission mode for each received packet, and (19) provisions a scheduling policy. This motivates us to develop a suboptimal cross-layer optimization algorithm.

D. Cross-Layer Optimization Algorithm

We now consider t as the time slot, whereas earlier, it denotes the iterative index. Let $\alpha^f_i = 1$. $\beta^f_{i,\,\mathrm{in}}(t)$ then reflects the size of input queue X^f_i at time slot t. $\sum_{\phi\in\Phi(i)} \gamma^f_{i,\,\phi}(t) + \gamma^f_i(t)$ denotes the number of packets leaving X^f_i at time t, whereas $\sum_{\phi\in\Phi(p^f(i)),\, f\in\phi} r_{p^f(i),\,\phi}(t) + r^f_{p^f(i)}(t)$ denotes the number of

packets coming at X_i^f at time t. Therefore, $\beta_{i,\text{in}}^f(t+1)$ calculated by (13) implies the size of X_i^f at time $t+1$. Similarly, $\gamma_{i,\phi}^f(t)$ denotes the number of packets of flow f arriving in $Z_{i,\phi}$ at time t, whereas $r_{i,\phi}(t)$ denotes the number of coded packets sent out at t. Therefore, $\beta_{i,\phi,\text{out}}^f(t)$ implies the size of $Z_{i,\phi}$ at time t. For the same reason, $\beta_{i,\text{out}}^f(t)$ reflects the size of W_i^f at time slot t.

There are three categories of variables in (17)–(19): the set of flow data rate $\vec{\lambda}$, transmission-mode assignment decision $\vec{\gamma}$, and transmission schedule $\vec{r}$. In each iterative step, the optimal solutions of these variables can be found using the Langrangian multipliers accordingly. In other words, (17)–(19) provide the policies for congestion control, transmission-mode assignment, and scheduling based on the current backlog of each queue.

We now proceed to describe our algorithm. At the beginning of each time slot, some new packets may arrive in X_i^f. Each node decides the transmission mode (traditionally or coded) for each packet in X_i^f such that the packets in X_i^f are transported to a certain output queue. Then, $p_f(i)$ can drop the packets that will be transmitted traditionally by i. After the transmission-mode assignment procedure, a certain feasible schedule is selected and then a set of links transmit at the current time slot. The transmission schedule at the current time slot would affect the queue size at each node at the next time slot, and then, the data rate injected into the network at the next time slot would be adjusted. In the next time slot, the whole procedure will be executed again. The procedures of flow control, transmission-mode assignment, and transmission scheduling cooperate with each other to improve network throughput performance. Each time slot is divided into three phases. In the first phase, the source node determines the amount of new packets injected into the network. In the second phase, node i decides the transmission mode (traditional or coded) of each newly received packet. This involves moving a packet from the input queue (X_i^f) to a certain output queue. In the third phase, a feasible schedule is selected to allow a set of links to transmit data packets during the whole time slot.

When the context is clear, we use $X_i^f(t)$ and $W_i^f(t)$ to denote the queue sizes at time slot t. Let $Z_{i,\phi}^f(t)$ denote the number of packets for flow f contained in queue $Z_{i,\phi}$ at time t. For instance, if $Z_{i,\phi}$ contains m coded packets and l native packets of f at time t, $Z_{i,\phi}^f(t) = m + l$, whereas $Z_{i,\phi}^{f'}(t) = m$, where $f' \in \phi$. In other words, $Z_{i,\phi}^f(t)$ is determined based on the current status of output queue $Z_{i,\phi}$. As aforementioned, we consider Langrangian multipliers in (13)–(15) as the queue sizes. We thus rewrite (17)–(19) as follows:

$$\lambda_f(t) = \arg\max_{0<\lambda_f\le\Lambda_f} U_f(\lambda_f) - \alpha_{s(f)}^f W_{s(f)}^f(t)\lambda_f \tag{20}$$

$$\vec{\gamma}(t) = \arg\max_{\vec{\gamma}\in\Gamma} \left\{ \sum_{i\in\mathcal{N},\phi\in\Phi(i),f\in\phi} \left(X_i^f(t) - Z_{i,\phi}^f(t)\right)\gamma_{i,\phi}^f + \sum_{f\in\mathcal{F},i\in\mathcal{N},i\neq s(f)} \left(X_i^f(t) - W_i^f(t)\right)\gamma_i^f \right\} \tag{21}$$

$$\vec{r}(t) = \arg\max_{\vec{r}\in\mathcal{R}} \left\{ \sum_{i,\phi\in\Phi(i)} \left(\sum_{f\in\phi} \left(Z_{i,\phi}^f(t) - X_{s^f(i)}^f(t)\right)\right) r_{i,\phi} + \sum_{i,f} \left(W_i^f(t) - X_{s^f(i)}^f(t)\right) r_i^f \right\}. \tag{22}$$

Equations (20)–(22) describe the policies on three layers: congestion control, transmission-mode assignment, and scheduling. Equation (20) specifies how to determine the data rate injected into the network in phase 1 of time slot t. Given $U_f(\lambda_f)$, we can calculate an optimal λ_f to maximize $U_f(\lambda_f) - \alpha_{s(f)}^f W_{s(f)}^f(t)\lambda_f$, In other words, (20) denotes a flow control policy, where $\alpha_{s(f)}^f$ is a factor for congestion control. $U_f(\lambda_f)$ is normally a monotonically increasing function of λ_f; thus, smaller $\alpha_{s(f)}^f$ implies that more packets would be injected into the network. $\alpha_{s(f)}^f$ should be set according to the buffer size. For instance, if each node in the network has a larger buffer size, we can set smaller $\alpha_{s(f)}^f$ to allow more packets injected into the network. On the other hand, the larger $\alpha_{s(f)}^f$ is suitable for the smaller buffer size of each node. According to (21), we should move the packet from X_i^f to the output queue containing the least number of packets for f. In the case that $Z_{i,\phi}$ and W_i^f contain the same number of packets for f, we prefer to move the packet to W_i^f. Equation (22) presents the scheduling policy to determine which links should transmit concurrently at time t. Based on our transmission-mode assignment scheme, $Z_{i,\phi}^f(t)$ must not be larger than $W_i^f(t)$. Under the stochastic traffic model, it is possible that $Z_{i,\phi}(t)$ contains all native packets of flow f but no packet of another flow f'. In this case, $(Z_{i,\phi}^f(t) - X_{s^f(i)}^f(t))r_{i,\phi} + (Z_{i,\phi}^{f'}(t) - X_{s^f(i)}^{f'}(t))r_{i,\phi}$ must not be larger than $(W_i^f(t) - X_{s^f(i)}^f(t))r_i^f + (W_i^{f'}(t) - X_{s^f(i)}^{f'}(t))r_i^f$, where $r_{i,\phi} = \min\{r_i^f, r_i^{f'}\}$. This implies that our scheduling scheme would not schedule a native packet in $Z_{i,\phi}$ to be transmitted. The joint solution of transmission-mode assignment and the scheduling policy intentionally assign some native packets in $Z_{i,\phi}$ waiting to be coded in the future. Moreover, the packets in $Z_{i,\phi}$ must be transmitted in a coded manner; the channel resources would be efficiently utilized. The outline of our algorithm is shown in Algorithm 1.

Algorithm 1 Cross-layer optimization algorithm

```
for Each time slot t do
    for each flow f going through i do
        if i = s(f) then
            if U(λ_f) − αW_i^f(t) · λ_f ≤ 0 for any λ_f ∈ (0, Λ_f] then
                set λ_f(t) = 0
            else
                inject new packets containing Λ_f amount of data.
        for each packet in X_i^f do
            move the packet to the output queue with the smallest size
```

10: Apply the greedy maximal scheduling (GMS)-framework to find a feasible schedule based on (22)

As mentioned in Section III, each vector $\vec{r}$ corresponds to a feasible schedule. Equation (22) represents a scheduling policy. We rewrite (22) as

$$M^*(t)=\arg\max_{M\in \mathrm{Co}(\mathcal{M})}\left\{\sum_{i,\,\phi\in\Phi(i)}\left(\sum_{f\in\phi}\left(Z^f_{i,\phi}(t)-X^f_{s^f(i)}(t)\right)\right)\times r_{i,\phi}(M)+\sum_{i,f}\left(W^f_i(t)-X^f_{s^f(i)}(t)\right)r^f_i(M)\right\}. \quad (23)$$

Finding the optimal solution of (23) is NP-hard [4]. Many practical scheduling solutions have been proposed. One of the most well-known suboptimal scheduling policies is the GMS policy [4]. GMS schedules links in decreasing order of the link weight conforming to interference constraints. According to the policy of (23), the weight of each link should correspond to the queue size and the rate of that link. Generally, given node i, the weight of each coding link in ϕ is

$$\omega(i,\,\phi)=\sum_{f\in\phi}\left(Z^f_{i,\phi}(t)-X^f_{s^f(i)}(t)\right)r_{i,\phi}. \quad (24)$$

We define the weight of each native link $(i,\,s^f(i))$ as

$$\omega^f_i=\left(W^f_i(t)-X^f_{s^f(i)}(t)\right)r^f_i. \quad (25)$$

Let $\mathbb{L}$ be the set of all the (native or coding) links with backlogged packets, and $\mathbb{L}'$ be the set of links to transmit at time t. At each time slot t, GMS selects the (native or coding) link with the maximum weight and moves the link from $\mathbb{L}$ to $\mathbb{L}'$. Then, GMS removes all the link interfering with the selected links in $\mathbb{L}'$. The process continues until $\mathbb{L}$ is empty. Our description assumes using the protocol interference model, i.e., the data rate on each link does not depend on the specific schedule. If we apply the physical-layer interference model, the weight on each link in $\mathbb{L}$ should be updated after each link is added into $\mathbb{L}'$. In [34], a distributed version for GMS, which is called LGMS, is developed. In our cross-layer optimization algorithm, flow control and transmission-mode assignment are performed by each node independently. Thus, our algorithm can be implemented in a distributed manner. In practical wireless networks, link rate and packet loss ratio are time-varying, as subjected to fading variations. Nevertheless, our scheduling policy can be easily extended to account for fading (see [5] for detailed discussion).

E. Stability Analysis

Earlier, we provide a suboptimal cross-layer optimization algorithm, developed by the joint of (20), (21), and (23). Without congestion control, the source node would accept all the packets from upper layer. Based on the proposed node model, (21) and (23) present the scheduling policy. The data rate region $\vec{\lambda}_\triangle$ supported by scheduling policy $\triangle$ means that, when the data rate strictly falls $\vec{\lambda}_\triangle$, scheduling policy $\triangle$ stabilizes the network. $\vec{\lambda}_\triangle$ must not be larger than the stability region Λ. A better scheduling policy would support a larger data rate region. In the following, we formally show that the data rate region supported by the proposed scheduling policy is Λ. We employ the quadratic Lyapunov function shown in the following:

$$L\left(\vec{X}(t),\vec{W}(t),\vec{Z}(t)\right)=\sum_{f,\,i\neq s(f)}\left(\left(X^f_i(t)\right)^2+\left(W^f_i(t)\right)^2+\sum_{\phi\in\Phi(i)}\left(Z^f_{i,\phi}(t)\right)^2\right)+\sum_f\left(W^f_{s(f)}(t)\right)^2. \quad (26)$$

Let $\lambda_f(t)$ denote the amount of data for flow f injected into the network. In the proposed policy, the queue update rules are as follows:

$$\begin{aligned}X^f_i(t+1)&=\left[X^f_i(t)-\left(\sum_{\phi\in\Phi(i)}\gamma^f_{i,\phi}(t)+\gamma^f_i(t)\right)\right]^+\\&\quad+\sum_{\phi'\in\Phi(j)}r^f_{j,\phi}+r^f_j(t),\quad j=p^f(i)\\W^f_i(t+1)&=\left[W^f_i(t)-r^f_i(t)\right]^++\gamma^f_i(t)\\Z^f_{i,\phi}(t+1)&=\left[Z^f_{i,\phi}(t)-r^f_{i,\phi}(t)\right]^++\gamma^f_{i,\phi}(t)\\W^f_{s(f)}(t+1)&=\left[W^f_{s(f)}(t)-r^f_{s(f)}(t)\right]^++\lambda_f(t).\end{aligned} \quad (27)$$

$r^f_{i,\phi}(t)$ denotes the amount of flow f transmitted by coding structure ϕ at time t, which depends on $r_{i,\phi}(t)$ and the backlogs in $Z_{i,\phi}$. Based on the transmission-mode assignment, if the packets in $Z_{i,\phi}$ are all native packets, ϕ would not be scheduled at t. That is, if $r_{i,\phi}(t)>0$, $Z_{i,\phi}$ must contain the coded packets from two flows. Therefore, we have

$$r^f_{i,\phi}(t)=r_{i,\phi}(t). \quad (28)$$

$X^f_i(t+1)$ is calculated as two parts: the remaining packets in X^f_i that are not transmitted at time slot t, and the amount of packets for f arriving at i at time slot t. Let each element in $\vec{\gamma}$ be upper bounded by a positive number. For instance, we set $\gamma^f_i(t)\le c_{(i,\,s^f(i))}$, where $c_{i,\,s^f(i)}$ is the maximum supported link rate between i and $s^f(i)$. Since each element in $\vec{r}$ and $\vec{\gamma}$ is upper bounded by a positive constant, according to in [35, Lemma 4.3] , we have the following:

$$\left(X^f_i(t+1)\right)^2-\left(X^f_i(t)\right)^2\le B_1+2X^f_i(t)\left(\left(\sum_{\phi'\in\Phi(j)}r^f_{j,\phi}+r^f_j(t)\right)-\left(\sum_{\phi\in\Phi(i)}\gamma^f_{i,\phi}(t)+\gamma^f_i(t)\right)\right)$$

$$\begin{aligned}
&\left(W_i^f(t+1)\right)^2 - \left(W_i^f(t)\right)^2 \\
&\quad \le B_2 + 2W_i^f(t)\left(\gamma_i^f(t) - r_i^f(t)\right) \\
&\left(Z_{i,\phi}^f(t+1)\right)^2 - \left(Z_{i,\phi}^f(t)\right)^2 \\
&\quad \le B_3 + 2Z_{i,\phi}^f(t)\left(\gamma_{i,\phi}^f(t) - r_{i,\phi}^f(t)\right) \\
&\left(W_{s(f)}^f(t+1)\right)^2 - \left(W_{s(f)}^f(t)\right)^2 \\
&\quad \le B_4 + 2W_i^f(t)\left(\lambda_f(t) - r_{s(f)}^f(t)\right) \qquad (29)
\end{aligned}$$

In (29), B_1, B_2, B_3, and B_4 are constant positive numbers. As $\vec{\lambda}$ strictly falls in the stability region Λ, there exists $\vec{\lambda}'$ also inside of Λ, such that each element in $\vec{\lambda}'$ is ϵ larger than the corresponding element in $\vec{\lambda}$, where ϵ is a small positive number. Denote by ν_i^f the rate of f carried on link $(i, s^f(i))$ and by $\nu_{i,\phi}^f$ the rate of the coded data carried in the coding structure ϕ at node i. We can identify ν_i^f and $\nu_{i,\phi}$ corresponding to $\vec{\lambda}'$; therefore, we have

$$\nu_{s(f)}^f = \lambda_f + \epsilon. \qquad (30)$$

According to (21) and (22), we have, respectively, the following:

$$\begin{aligned}
&\left\{\sum_{i,f,\phi\in\Phi(i)}\left(X_i^f(t) - Z_{i,\phi}^f(t)\right)\nu_{i,\phi}\right. \\
&\quad \left. + \sum_{i\ne s(f),f}\left(X_i^f(t) - W_i^f(t)\right)\nu_i^f\right\} \\
&\quad \le \left\{\sum_{i,f,\phi\in\Phi(i)}\left(X_i^f(t) - Z_{i,\phi}^f(t)\right)\gamma_{i,\phi}^f(t)\right. \\
&\qquad \left. + \sum_{i\ne s(f),f}\left(X_i^f(t) - W_i^f(t)\right)\gamma_i^f(t)\right\} \qquad (31) \\
&\left\{\sum_{i,\phi\in\Phi i}\left(\sum_{f\in\phi}\left(Z_{i,\phi}^f(t) - X_{s^f(i)}^f(t)\right)\right)\nu_{i,\phi}\right. \\
&\quad \left. + \sum_{i,f}\left(W_i^f(t) - X_{s^f(i)}^f(t)\right)\nu_i^f\right\} \\
&\quad \le \left\{\sum_{i,\phi\in\Phi i}\left(\sum_{f\in\phi}\left(Z_{i,\phi}^f(t) - X_{s^f(i)}^f(t)\right)\right)r_{i,\phi}(t)\right. \\
&\qquad \left. + \sum_{i,f}\left(W_i^f(t) - X_{s^f(i)}^f(t)\right)r_i^f(t)\right\}. \qquad (32)
\end{aligned}$$

We then calculate (33), shown below. By (29), we have the inequality (a), where B is a constant positive number. Based on (28), we have equality (b). By (31) and (32), we have inequality (c) in (33). Since $X_{d(f)}^f(t) = 0$ for any t, where $d(f)$ is the destination of f, we have equality (d).

$$\begin{aligned}
&L\left(\vec{X}(t+1), \vec{W}(t+1), \vec{Z}(t+1)\right) - L\left(\vec{X}(t), \vec{W}(t), \vec{Z}(t)\right) \\
&\quad \le_{(a)} B + 2\sum_{f,\, i\ne s(f)} \\
&\qquad \times \left\{X_i^f(t)\left(\left(\sum_{j=p^f(i),\phi'\in\Phi(j)} r_{j,\phi}^f + r_j^f(t)\right)\right.\right. \\
&\qquad\qquad \left.\left. - \left(\sum_{\phi\in\Phi(i)}\gamma_{i,\phi}^f(t) + \gamma_i^f(t)\right)\right)\right. \\
&\qquad\quad + W_i^f(t)\left(\gamma_i^f(t) - r_i^f(t)\right) \\
&\qquad\quad \left. + \sum_{\phi\in\Phi(i)} Z_{i,\phi}^f(t)\left(\gamma_{i,\phi}^f(t) - r_{i,\phi}^f(t)\right)\right\} \\
&\qquad + 2\sum_f W_{s(f)}^f(t)\left(\lambda_f(t) - r_{s(f)}^f(t)\right) \\
&\quad =_{(b)} B + 2\sum_f W_{s(t)}^f(t)\lambda_f(t) \\
&\qquad - 2\left\{\sum_{i,\phi\in\Phi i}\left(\sum_{f\in\phi}\left(Z_{i,\phi}^f(t) - X_{s^f(i)}^f(t)\right)\right)r_{i,\phi}(t)\right. \\
&\qquad\quad \left. + \sum_{i,f}\left(W_i^f(t) - X_{s^f(i)}^f(t)\right)r_i^f(t)\right\} \\
&\qquad - 2\left\{\sum_{i,f,\phi\in\Phi(i)}\left(X_i^f(t) - Z_{i,\phi}^f(t)\right)\gamma_{i,\phi}^f(t)\right. \\
&\qquad\quad \left. + \sum_{i\ne s(f),f}\left(X_i^f(t) - W_i^f(t)\right)\gamma_i^f(t)\right\} \\
&\quad \le_{(c)} B + 2\sum_f W_{s(t)}^f(t)\lambda_f(t) \\
&\qquad - 2\left\{\sum_{i,\phi\in\Phi i}\left(\sum_{f\in\phi}\left(Z_{i,\phi}^f(t) - X_{s^f(i)}^f(t)\right)\right)\nu_{i,\phi}\right. \\
&\qquad\quad \left. + \sum_{i,f}\left(W_i^f(t) - X_{s^f(i)}^f(t)\right)\nu_i^f\right\} \\
&\qquad - 2\left\{\sum_{i,f,\phi\in\Phi(i)}\left(X_i^f(t) - Z_{i,\phi}^f(t)\right)\nu_{i,\phi}\right. \\
&\qquad\quad \left. + \sum_{i\ne s(f),f}\left(X_i^f(t) - W_i^f(t)\right)\nu_i^f\right\} \\
&\quad =_{(d)} B + 2\sum_f W_{s(t)}^f(t)\lambda_f(t) \\
&\qquad - 2\left\{\sum_f W_{s(f)}^f\nu_{s(f)}^f + \sum_f X_{s^f(s(f))}^f(t)\nu_{i,\phi}\right\} \\
&\quad \le B + 2\sum_f W_{s(t)}^f(t)\lambda_f(t) - 2\sum_f W_{s(f)}^f\nu_{s(f)}^f \qquad (33)
\end{aligned}$$

$$\begin{aligned}\Delta(t) &\triangleq \mathbb{E}\left\{L\left(\vec{X}(t+1),\vec{W}(t+1),\vec{Z}(t+1)\right)\right.\\ &\qquad \left.-L\left(\vec{X}(t),\vec{W}(t),\vec{Z}(t)\right)\middle|L\left(\vec{X}(t),\vec{W}(t),\vec{Z}(t)\right)\right\}\\ &\le B+2\sum_f W^f_{s(t)}(t)\lambda_f-2\sum_f W^f_{s(f)}\nu^f_{s(f)}\\ &= B-2\sum_f W^f_{s(f)}\epsilon\,[\text{based on (30)}]\,. \end{aligned} \tag{34}$$

Finally, we calculate the Lyapunov drift as in (34). We have shown that our policy satisfies the conditions of [35, Lemma 4.2], and therefore, our policy stabilizes the network for all arrival rate vectors that are strictly interior to the stability region.

V. Simulation Results

Here, we evaluate the performance of our proposed algorithms via packet-level simulation. We compare the proposed algorithm with the method [2], [5] presented in (1). The work in [8] describes the disparity in the buffer size used in various research platforms. The buffer keeps both the packets waiting to be transmitted and the packets waiting for decoding future coded packets. If the buffer at a node is full, the node will drop the new incoming packet in default [7]. Another option is to drop a sent packet to accommodate the new incoming packet. If the sent packet dropped is needed for decoding later, more bandwidth resources will be wasted. Therefore, dropping new incoming packets is more appropriate. For simplicity, each link in the network has the same transmission rate of 1 Mb/s. In our simulation, we use the objective of maximizing network throughput, i.e., $U(\lambda_f)=\lambda_f$. α in Algorithm 1 is a factor for congestion control. We apply the commonly used 2-hop interference model [4] in our simulations. Each second is divided into 100 time slots, and each link can transmit a packet in each time slot. The simulation runs 500 s, and the average throughput is counted as the amount of data received by the destinations divided by 500 s.

We randomly deploy 80 nodes in an area of 1000 m $*$ 1000 m. The transmission range is 200 m. We find 12 source–destination pairs, and then deploy two flows with opposite direction on each pair. There are a total of 24 flows in the network. For each pair of a source and a destination, the average data rate of one direction is twice of that of the other direction. Here, the data rate means the amount of data injected from the upper layer, and the amount of data getting into a routing layer depends on the specific congestion control scheme. We conduct simulations on two data arrival modes: Poisson and uniform. We produce different instances for each scenario, and the results are the average on five different instances. We first apply the default ns-2 buffer size of 50 packets in each node. We set $\alpha=0.1$, which implies that the source node will not inject new data packets into the network if the buffer size is larger than 10, which is called the congestion window size.

Fig. 4 shows the throughput and packet loss ratio with varying the average data rate of the flows. The data arrival follows Poisson distribution. When the data rate is small, the network can support all the injected packets; thus, the throughput with the existing method differs only a little with the proposed

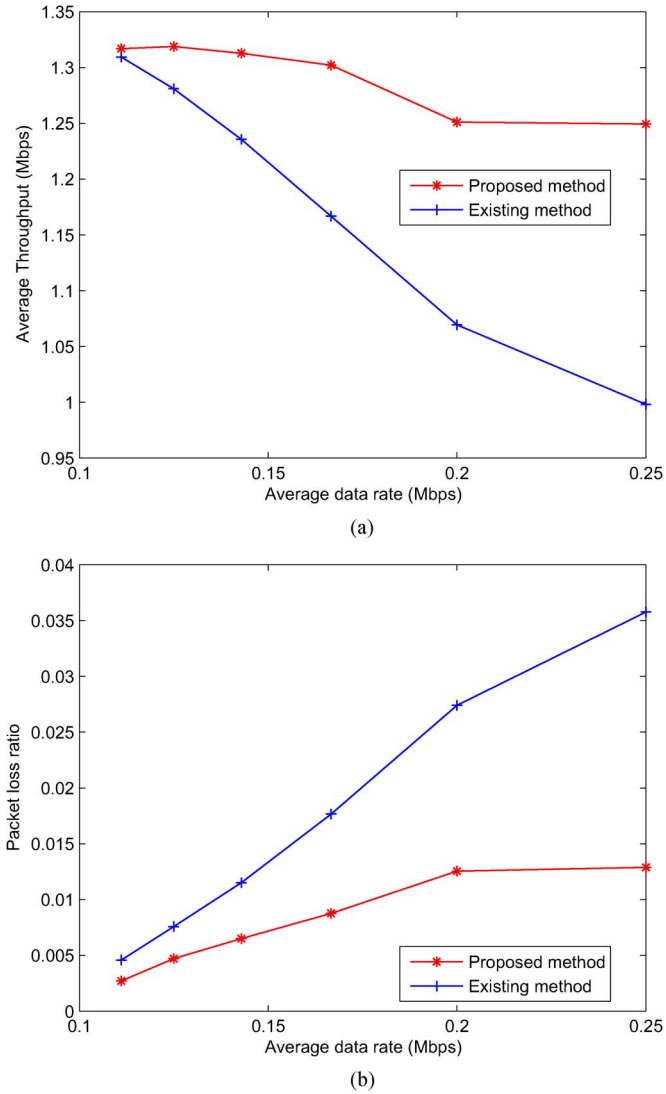


Fig. 4. Poisson arrival. (a) Average throughput. (b) Average packet loss ratio.

method, as shown in Fig. 4(a). As the average data rate increases, some packets would be dropped due to buffer overflow. Since the existing method requires a larger decoding buffer, the throughput decreases due to packet dropping. Although the throughput of the proposed method also reduces, the gap between the throughputs of the two methods becomes larger as the data rate increases. In Fig. 4(a), when the data rate changes from 0.2 to 0.25, the throughput of our policy is slightly reduced. We apply the same congestion control policy in the existing scheduling mechanism. With the congestion control, the total number of packets injected into the network is almost the same. Thus, the throughput does not change a lot. This implies that, when the data rate is too large, congestion control would guarantee the stable throughput of the network. We also observe that the stable throughput of the existing method is much lower than that of our method. Fig. 4(b) shows the change of packet loss ratio as the data rate varies. Note that packet loss means that packets are dropped due to overflow. As the data rate increases, the packet loss ratio increases, and the packet loss ratio of the existing method grows faster than that of our method. We observe that the packet loss ratio of our

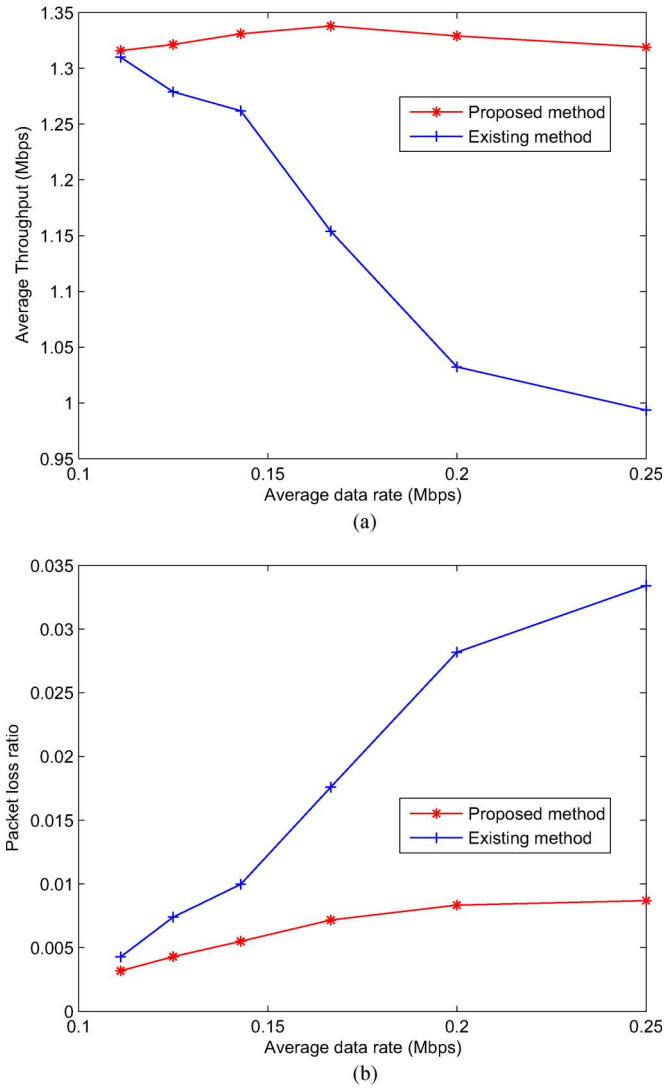


Fig. 5. Uniform arrival. (a) Average throughput. (b) Average packet loss ratio.

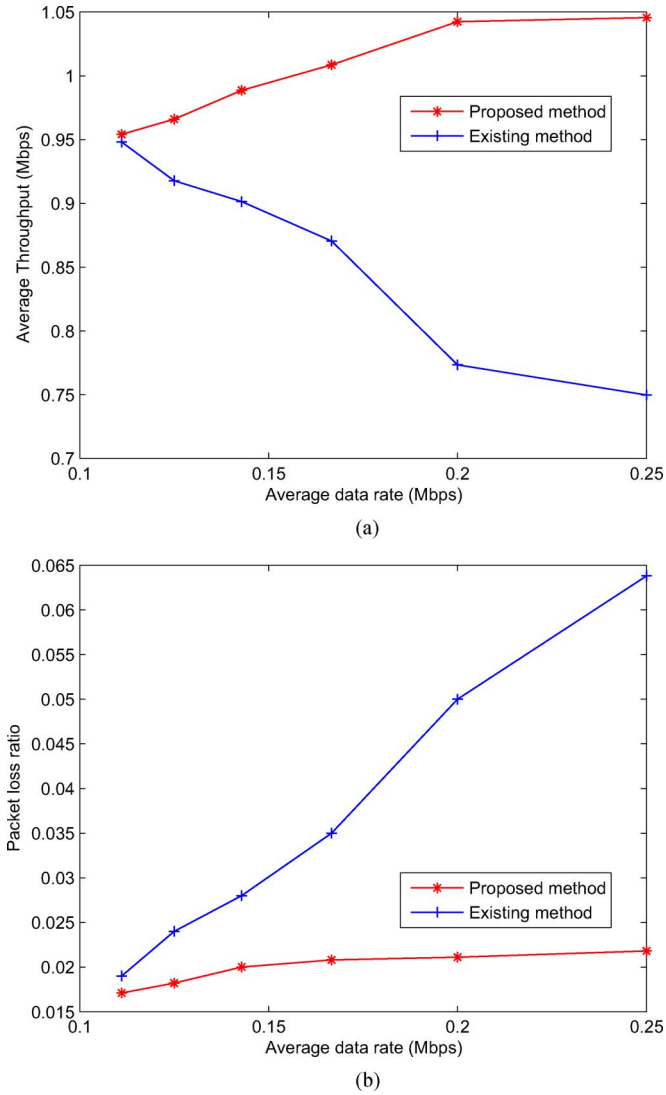


Fig. 6. Rayleigh fading channel. (a) Average throughput. (b) Average packet loss ratio.

proposed method does not grow a lot when the data rate is larger than 0.2, which is because the number of packets injected into the network does not increase by using congestion control. Fig. 5 shows the simulation results with the uniform data arrival distribution, and the performance of our algorithm is similar as that with the Poisson data arrival model.

In both Figs. 4 and 5, channel is assumed to be ideal, and each packet transmission is successful. We then simulate the Rayleigh fading channel and evaluate the performance of the two methods. We apply the Poisson arrival model, and Fig. 6 shows the simulation results. We observe the different variation trends for the throughput of the two algorithms in Fig. 6(a). When the data rate is small, the network is not saturated, the packet loss is mainly due to fading. Therefore, the throughputs and packet loss ratios of two algorithms are almost the same. Since some packets are lost due to fading, as compared with the links without fading, buffer overflow occurs at a higher data rate. In other words, a larger data rate is required to saturate the network when considering fading. This explains that the throughput of our algorithm with fading is lower than that without fading. Moreover, the maximum throughput of our algorithm with fading happens at a larger data rate as compared with that without fading. On the other hand, the network is saturated at a small data rate with the existing method. Therefore, the throughput decreases as the data rate increases, and the packet loss at the large data rate is mainly due to buffer overflow. Fig. 6(b) verified that the packet loss ratio of our algorithms grows a little bit. Since throughput of our policy increases a little bit as data rate increases, the number of packets being transmitted increases a little. Moreover, the packet loss is mainly due to fading under this situation; thus, the packet loss ratio grows slightly. Afterward, we let the successful transmission ratio on each link randomly fall in [0.85, 0.99], and Fig. 7 shows the simulation results. For the same reason, the performance change is similar as that with the Rayleigh fading channel.

Afterward, we study the performance when the links are of different rates. We let the transmission rate of each link follow the uniform distribution [1, 10] Mb/s. Since the transmission rate is larger, we set the buffer size of each node to be

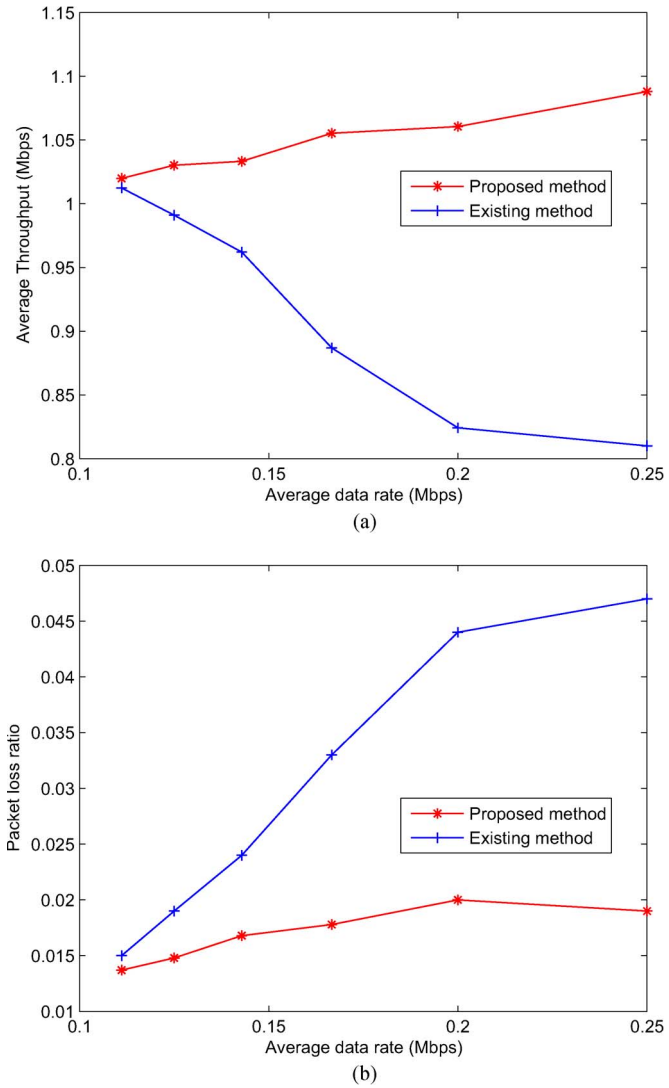


Fig. 7. Links with successful transmission ratio between [0.85, 0.99]. (a) Average throughput. (b) Average packet loss ratio.

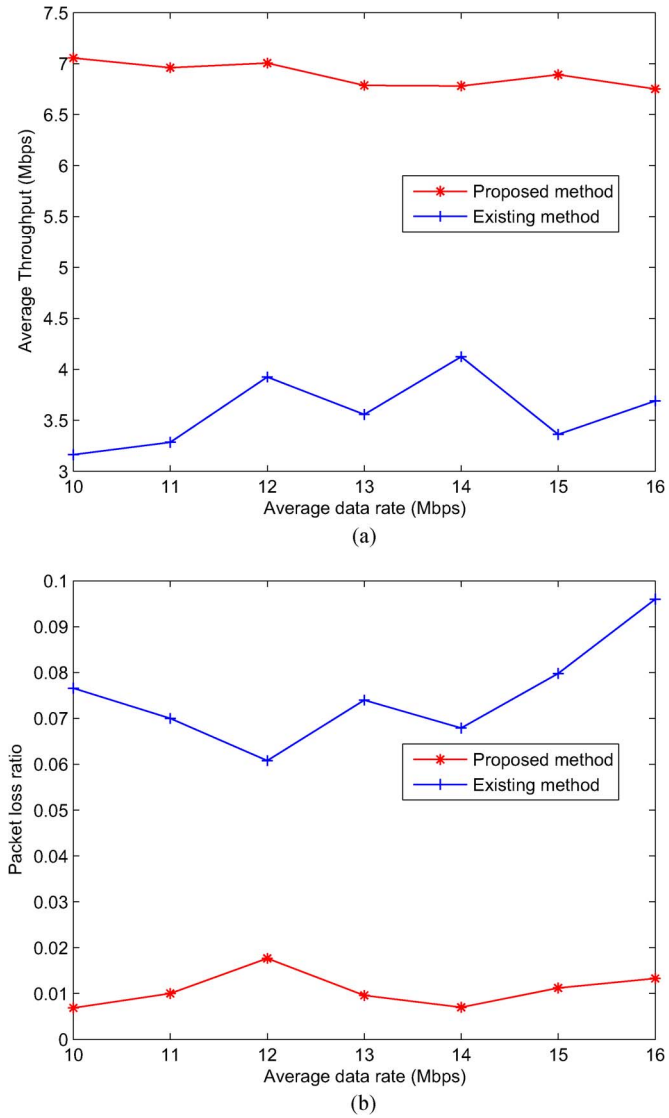


Fig. 8. Multirate links. (a) Average throughput. (b) Average packet loss ratio.

1000 packets and the congestion control window size to be 200 packets. Fig. 8 shows the simulation results. Performance analysis in a multirate scenario is more complicated than that in single rate. When the data rate is small, the backlog may be smaller than the transmission rate; therefore, bandwidth resources would not be fully utilized. On the other hand, the larger data rate may introduce more packets dropped, which reduces network throughput. This is why we observe that the throughput does not have an obvious trend. Generally, our algorithm yields higher throughput and lower packet loss ratio.

We also test the performance of our algorithm as the buffer size and the congestion control window size change. When the number of packets in the output buffer of the source node is larger than the congestion window size, the source node would not inject new packets from the upper layer. The data rate is set to 0.25 Mb/s. We set the congestion window size to be 20 packets, and Fig. 9 shows the simulation results with a change of buffer size. As fewer packets would be dropped by using a larger buffer size, the throughput increases as the buffer size increases. When the buffer size is large enough, there are almost no packets dropped; therefore, the throughput of our algorithm is almost the same as that of the existing algorithm. We observe that the throughput of our algorithm is much larger than that of the existing algorithm at several instances. This shows that buffer space plays a more significant impact on the existing algorithm than our algorithm. We then set the buffer size to be 100 packets, and Fig. 10 shows the simulation results as congestion window size changes. When the congestion window size is smaller, less packets would be injected into the network, so that the network would not be heavy loaded and less packets are dropped due to overflow. Therefore, the throughput is higher, and packet loss ratio is lower when the congestion window size is smaller. With the larger congestion control window size, more packets are injected into the network. The existing algorithm requires more buffer space than our algorithm; therefore, more packets are dropped. We thus observe that the throughput of the existing algorithm reduces more quickly than that of our algorithm.

Generally, under the limited buffer space, the proposed algorithm can effectively reduce buffer overhead introduced by

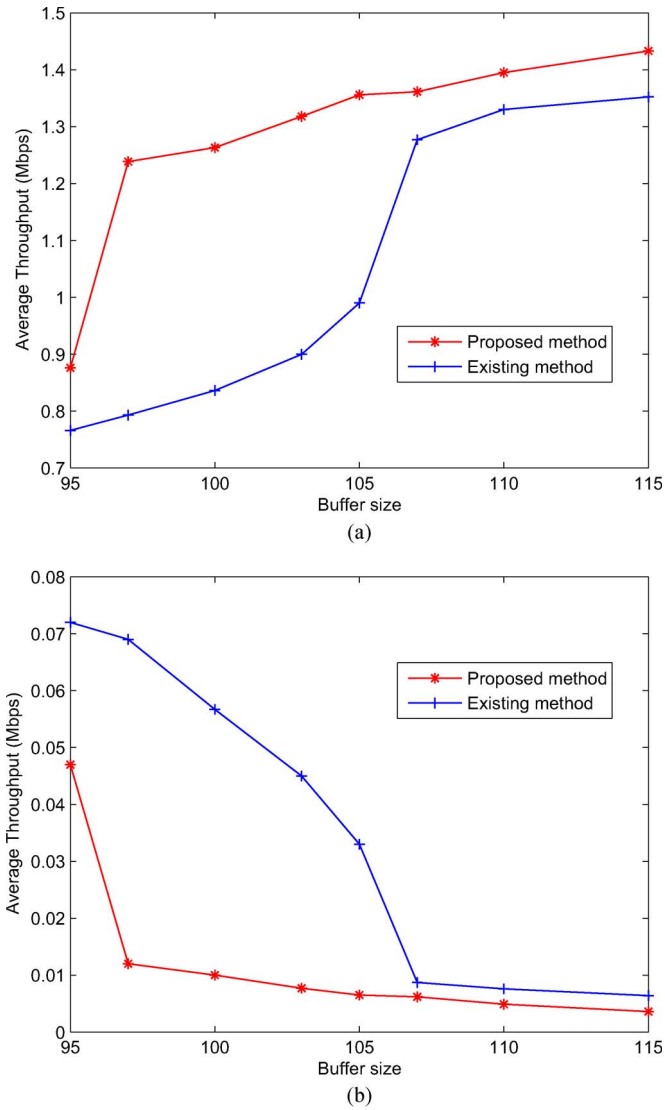


Fig. 9. Impact of buffer size. (a) Average throughput. (b) Average packet loss ratio.

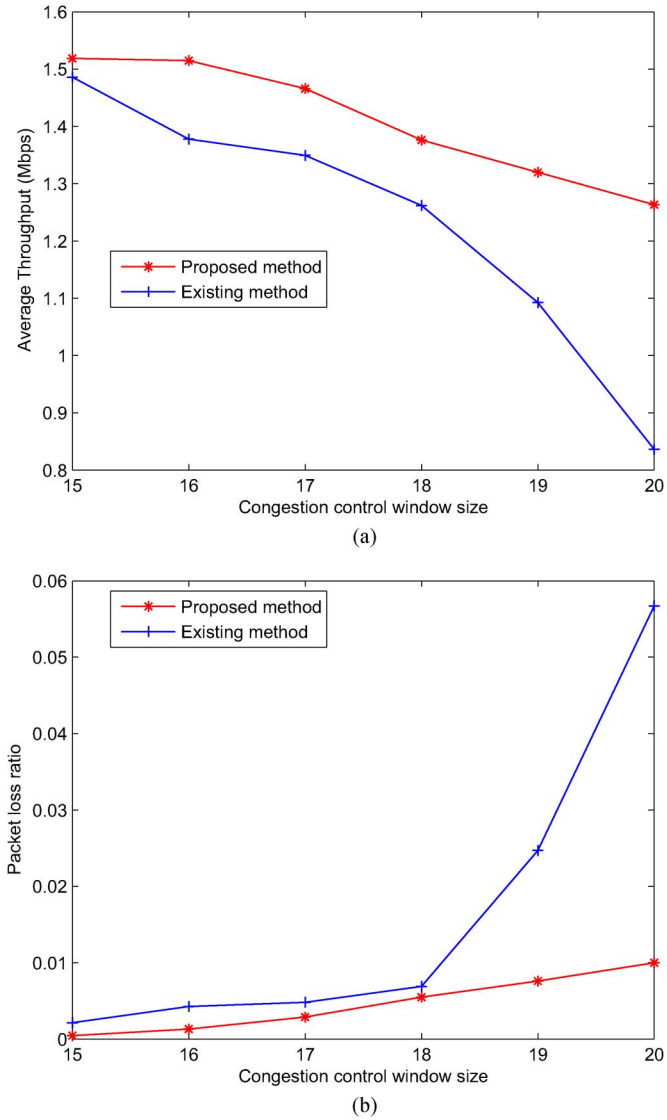


Fig. 10. Impact of congestion control window size. (a) Average throughput. (b) Average packet loss ratio.

network coding through the transmission-mode preassignment procedure. In particular, under the heavy-loaded network, the proposed algorithm produces higher throughput and smaller packet loss ratio compared with the existing algorithm.

VI. Conclusion

This paper has studied a scheduling issue with network coding in wireless networks. In particular, the space overhead issue introduced by network coding has been studied since each node normally has a finite buffer space, and packets may be dropped due to overflow. To reduce packet loss ratio and improve network throughput, this paper has proposed a scheduling scheme comprising a transmission-mode preassignment procedure and a transmission scheduling procedure. Simulation results demonstrated that space overhead introduced by network coding significantly affects network performance, and the proposed scheduling method outperforms the existing method.

References

[1] A. Khreishah, C.-C. Wang, and N. B. Shroff, "Cross-layer optimization for wireless multihop networks with pairwise intersession network coding," *IEEE J. Sel. Areas Commun.*, vol. 27, no. 5, pp. 606–621, Jun. 2009.

[2] N. M. Jones, B. Shrader, and E. Modiano, "Optimal routing and scheduling for a simple network coding scheme," in *Proc. IEEE INFOCOM*, Mar. 2012, pp. 352–360.

[3] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Medard, and J. Crowcroft, "Xors in the air: Practical wireless network coding," *IEEE/ACM Trans. Netw.*, vol. 16, no. 3, pp. 497–510, Jun. 2008.

[4] C. Joo and N. B. Shroff, "Local greedy approximation for scheduling in multi-hop wireless networks," *IEEE Trans. Mobile Comput.*, vol. 11, no. 3, pp. 414–426, Mar. 2012.

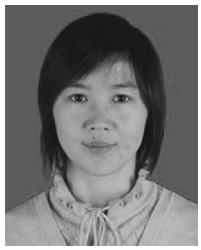
[5] P. Chaporkar and A. Proutiere, "Adaptive network coding and scheduling for maximizing throughput in wireless networks," in *Proc. ACM MOBICOM*, Sep. 2007, pp. 135–146.

[6] B. Lorenzo and S. Glisic, "Optimal routing and traffic scheduling for multihop cellular networks using genetic algorithm," *IEEE Trans. Mobile Comput.*, vol. 12, no. 11, pp. 2274–2288, Nov. 2013.

[7] A. Krifa, C. Barakaty, and T. Spyropoulos, "Optimal buffer management policies for delay tolerant networks," *Proc. IEEE SECON*, pp. 260–268, Jun. 2008.

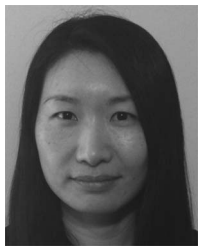
[8] K. Jamshaid, B. Shihada, L. Xia, and P. Levis, "Buffer sizing in 802.11 wireless mesh networks," *Proc. IEEE MASS*, pp. 272–281, 2011.

[9] T. Li, D. Leith, and D. Malone, "Buffer sizing for 802.11-based networks," *IEEE/ACM Trans. Netw.*, vol. 19, no. 1, pp. 156–169, Feb. 2011.
[10] M. Chiang, S. H. Low, A. R. Calderbank, and J. C. Doyle, "Layering as optimization decomposition: A mathematical theory of network architectures," *Proc. IEEE*, vol. 95, no. 1, pp. 255–312, Jan. 2007.
[11] X. Lin and N. B. Shroff, "The impact of imperfect scheduling on cross-layer congestion control in wireless networks," *IEEE/ACM Trans. Netw.*, vol. 14, no. 2, pp. 302–315, Apr. 2006.
[12] A. Eryilmaz and R. Srikant, "Fair resource allocation in wireless networks using queue-length based scheduling and congestion control," *IEEE/ACM Trans. Netw.*, vol. 15, no. 6, pp. 1333–1344, Dec. 2007.
[13] L. Bui, A. Eryilmaz, R. Srikant, and X. Wu, "Joint asynchronous congestion control and distributed scheduling for multi-hop wireless networks," in *Proc. IEEE INFOCOM*, Apr. 2006, pp. 1–12.
[14] K. Ronasi, V. Wong, and S. Gopalakrishnan, "Distributed scheduling in multihop wireless networks with maxmin fairness provisioning," *IEEE Trans. Wireless Commun.*, vol. 11, no. 5, pp. 1753–1763, May 2012.
[15] M. Alicherry, R. Bhatia, and E. Li, "Joint channel assignment and routing for throughput optimization in multi-radio wireless mesh networks," in *Proc. ACM MobiCom*, Aug. 2005, pp. 58–72.
[16] M. Kodialam and T. Nandagopal, "Characterizing the capacity region in multi-radio multi-channel wireless mesh networks," in *Proc. ACM Mobicom*, Aug. 2005, pp. 73–87.
[17] X. Lin and S. Rasool, "A distributed joint channel-assignment, scheduling and routing algorithm for multi-channel ad hoc wireless networks," in *Proc. IEEE INFOCOM*, May 2007, pp. 1118–1126.
[18] S. Merlin, N. H. Vaidya, and M. Zorzi, "Resource allocation in multi-radio multi-channel multi-hop wireless networks," in *Proc. IEEE INFOCOM*, Apr. 2008, pp. 610–618.
[19] H. Li, Y. Cheng, X. Tian, and X. Wang, "A generic framework for throughput-optimal control in MR-MC wireless networks," in *Proc. IEEE INFOCOM*, Apr. 2012, pp. 145–153.
[20] H. Yomo and P. Popovski, "Opportunistic scheduling for wireless network coding," in *Proc. IEEE ICC*, Jun. 2007, pp. 5610–5615.
[21] W. Chen, K. B. Letaief, and Z. Cao, "Buffer-aware network coding for wireless networks," *IEEE/ACM Trans. Netw.*, vol. 20, no. 5, pp. 1389–1401, Oct. 2012.
[22] D. Traskov, M. Medard, P. Sadeghi, and R. Koetter, "Joint scheduling and instantaneously decodable network coding," in *Proc. IEEE GLOBECOM*, Dec. 2009, pp. 1–6.
[23] B.-G. Kim and J.-W. Lee, "Opportunistic resource scheduling for ofdma networks with network coding at relay stations," *IEEE Trans. Wireless Commun.*, vol. 11, no. 1, pp. 210–221, Jan. 2012.
[24] Y. E. Sagduyu, R. A. Berry, and D. Guo, "Throughput and stability for relay-assisted wireless broadcast with network coding," *IEEE J. Sel. Areas Commun.*, vol. 31, no. 8, pp. 1506–1516, Aug. 2013.
[25] J. Le, C.-S. Lui, and D.-M. Chiu, "How many packets can we encode?- An analysis of practical wireless network coding," in *Proc. IEEE INFOCOM*, Apr. 2008, pp. 1040–1048.
[26] J. Liu, D. Goeckel, and D. Towsley, "Bounds on the throughput gain of network coding in unicast and multicast wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 27, no. 5, pp. 582–592, Jun. 2009.
[27] S. Sengupta, S. Rayanchu, and S. Banerjee, "Network coding-aware routing in wireless networks," *Proc. IEEE/ACM Trans. Netw.*, vol. 18, no. 4, pp. 1158–1170, Aug. 2010.
[28] T. Cui, L. Chen, and T. Ho, "Energy efficient opportunistic network coding for wireless networks," in *Proc. IEEE INFOCOM*, Apr. 2008, pp. 1022–1030.
[29] J. Zhang and P. Fan, "Optimal scheduling for network coding: Delay v.s. efficiency," in *Proc. IEEE GLOBECOM*, Dec. 2010, pp. 1–5.
[30] V. J. Venkataramanan and X. Lin, "Low-complexity scheduling algorithm for sum-queue minimization in wireless convergecast," in *Proc. IEEE INFOCOM*, Apr. 2011, pp. 2336–2344.
[31] B. Ji, C. Joo, and N. B. Shroff, "Delay-based back-pressure scheduling in multi-hop wireless networks," in *Proc. IEEE INFOCOM*, Apr. 2011, pp. 2579–2587.
[32] Y. Shi, Y. T. Hou, J. Liu, and S. Kompella, "Bridging the gap between protocol and physical models for wireless networks," *IEEE Trans. Mobile Comput.*, vol. 12, no. 7, pp. 1404–1406, Jul. 2013.
[33] L. Ying, S. Shakkottai, and A. Reddy, "On combining shortest-path and back-pressure routing over multihop wireless networks," *IEEE/ACM Trans. Netw.*, vol. 19, no. 3, pp. 841–854, Jun. 2011.
[34] M. Leconte, J. Ni, and R. Srikant, "Improved bounds on the throughput efficiency of greedy maximal scheduling in wireless networks," in *Proc. ACM MOBIHOC*, 2009, pp. 165–174.
[35] L. Tassiulas, L. Georgiadis, and M. J. Neely, *Resource Allocation and Cross-Layer Control in Wireless Networks*. Delft, The Netherlands: Now, 2006.



Ronghui Hou received the B.Eng., M.Eng., and Ph.D. degrees in communication engineering from Northwestern Polytechnical University, Xi'an, China, in 2002, 2005, and 2007, respectively.

From 2007 to 2009, she was a Postdoctoral Fellow with the Department of Electrical and Electronic Engineering, The University of Hong Kong, Kowloon, Hong Kong. Since December 2009, she has been with Xidian University, Xi'an, China, where she is currently an Associate Professor with the Department of Telecommunication Engineering. Her research interests include network quality-of-service issues, routing algorithm design, and wireless networks.



King-Shan Lui (S'00–M'03–SM'14) received the Ph.D. degree in computer science from the University of Illinois at Urbana–Champaign, Champaign, IL, USA.

She is currently an Associate Professor with the Department of Electrical and Electronic Engineering, The University of Hong Kong, Kowloon, Hong Kong. Her research interests include network protocols design and analysis, wireless networks, smart grids, and quality-of-service issues.

Jiandong Li (SM'05) received the Bachelor's, Master's, and Ph.D. degrees in communications and electronic systems from Xidian University, Xi'an, China, in 1982, 1985, and 1991, respectively.

Since 1985, he has been with Xidian University and became a Professor in 1994 and the Dean of the School of Telecommunication Engineering in 1997. From January 2002 to January 2003, he was a Visiting Professor with the School of Electrical and Computer Engineering, Cornell University, Ithaca, NY, USA. His current research interests include wireless communications, network protocols, and algorithm design.