Maximizing Throughput in Wireless Networks with Finite Internal Buffers

Ching-Min Lien, Cheng-Shang Chang, Jay Cheng and Duan-Shin Lee
Institute of Communications Engineering
National Tsing Hua University
Hsinchu 300, Taiwan, R.O.C.
E-mail: keiichi@gibbs.ee.nthu.edu.tw; cschang@ee.nthu.edu.tw;
jcheng@ee.nthu.edu.tw; lds@cs.nthu.edu.tw

ABSTRACT

In this paper, we consider the problem for maximizing the throughput of a discrete-time *wireless* network, where only certain sets of links can transmit simultaneously. It is well-known that each set of such links can be represented by a configuration vector and the convex hull of the configuration vectors determines the capacity region of the wireless network. In the literature, packet scheduling polices that stabilize any admissible traffic in the capacity region are mostly related to the maximum weighted matching algorithm (MWM) that identifies the most suitable configuration vector in every time slot. Unlike the MWM algorithm, we propose a dynamic frame sizing (DFS) algorithm that also stabilizes any admissible traffic in the capacity region. The DFS algorithm, as an extension of our previous work for *wired* networks, also does not have a fixed frame size. To determine the frame size, an optimization problem needs to be solved at the beginning of each frame. Once the frame size is determined, a hierarchical smooth schedule is devised to determine both the schedule for configuration vectors and the schedule for multicast traffic flows in each link. Under the assumption of Bernoulli arrival processes with admissible rates, we show that the number of packets of each multicast traffic flow inside the wireless network is bounded above by a constant and thus one only requires to implement a finite internal buffer in each link in such a wireless network.

I. Introduction

Packet scheduling in both wired and wireless networks to achieve maximum system throughout or provide quality of service has been an ongoing research problem for a long period of time. Our objective in this paper is to extend the dynamic frame sizing (DFS) algorithm for switches [3] and wired networks [8] to the setting of wireless networks. For this, we consider the *configuration vector* model that is often used in the literature to model the effect of link interference in a wireless network. A configuration vector is a vector of indicator variables that specifies a set of links which are allowed to transmit packets at the same time in a wireless network. The configuration vector model then characterizes a wireless network by a set of configuration vectors. Such a model is also known as the generalized constrained queueing model in [4]. For the configuration vector model, it is well-known that the capacity region for a wireless network is the convex hull of the set of configuration vectors. There are plenty of studies in the literature (see e.g., [13], [12], [4], [7]) that addressed the packet scheduling problem for maximizing the throughput in such a wireless network model. In particular, a scheduling algorithm is called *throughput-optimal* if it can stabilize the network for arrival traffic with rates falling within the capacity region. Most of throughput-optimal scheduling schemes are related to the maximum weighted matching (MWM) algorithm in [13], which identifies the most suitable configuration vector according to the queue length information available at each time slot.

Unlike the MWM algorithm, there is no need for the DFS algorithm to solve an optimization problem in every time slot. In the DFS algorithm, time is partitioned into frames, and an optimization problem is solved to determine the frame size at the beginning of each frame. For a wireless network that implements per flow queueing for each multicast flow, the frame size is chosen to be the minimum amount of time, known as the minimum clearance time, to clear the backlogs observed at the ingress queues at the beginning of the frame. By so doing, the backlogs at the ingress queues at the beginning of a frame is then bounded above by the arrivals during the previous frame, and a packet that arrives at an ingress queue in one frame will leave the ingress queue and enter the network in the next frame. Thus, as long as the expected size of each frame is finite, the expected backlog at each ingress queue remains finite. For packets that have departed from their ingress queues and entered the network, the DFS algorithm provides each flow in a frame a guaranteed rate that is proportional to the backlog observed at the ingress queue at the beginning of that frame. Such a guaranteed rate service then ensures the number of packets of each flow inside the network is bounded above by a finite constant. As a result, every internal buffer is finite and this mitigates the problem of implementing unlimited internal buffers as pointed out in [5].

The extension of the DFS algorithm to the configuration vector model is not as straightforward as one might expect. There are two technical difficulties that need to be conquered

for such an extension. First, unlike the models for switches [3] and wired networks [8], there is no explicit expression for the frame size in the configuration model as the minimum clearance time is now a solution of an optimization problem. Without an explicit expression for the frame size, it would be difficult to use the large deviation argument in [3], [8] to prove the finiteness of the expected frame size. Second, packet scheduling in the configuration vector model is much more complicated than that for switches and wired networks. In the configuration vector model, there are two levels of scheduling: (i) the upper level for scheduling configuration vectors and (ii) the lower level for scheduling flows in each link. Providing guaranteed rate services in such a two-level schedule for each frame poses a challenging problem.

Our contributions of this paper are mainly to solve these two technical problems. For the problem on the frame size, we derive an explicit upper bound for the minimum clearance time that can be used for the large deviation argument. Our idea for the upper bound is to compare the minimum clearance time with the time to drain the backlog at each link with the rate equal to its arrival rate (even though the actual arrival rate is *not* known). By so doing, we are able to establish the connection between the external arrival rates and the frame size so that the large deviation argument in [3], [8] can be used. To provide guaranteed rate services inside the network, we propose a hierarchical smooth schedule (as an extension of the smooth schedule in [6], [3], [8]). We then derive bounds on the differences between the guaranteed rate services provided by the hierarchical smooth schedule and the ideal rate services. These bounds are then used for bounding the number of packets in each internal queue.

The rest of this paper is organized as follows. First, we describe our mathematical model in Section II. In Section III, we propose our DFS algorithm, including determining the frame size in Section III-A, proposing the hierarchical smooth schedule in Section III-B, and proving the finiteness of the expected frame size in Section III-C. In Section IV, we conclude this paper by addressing some further extensions.

II. The mathematical model

In this section, we first introduce the mathematical model and the notations that will be used in the paper.

A. Per flow queueing for multicast flows

Consider a network with L links and J multicast traffic flows, indexed from 1 to L and from 1 to J, respectively. Each flow enters the network at some queue, traverses through a set of queues arranged in a fan-out tree, and then leaves the network. There might be several flows traversing a common link, and we assume that per flow queueing is used in every link, i.e., every flow has its own queue in every link it traverses. We define $q_\ell^{(j)}$ as the queue for flow j traversing link ℓ. Moreover, we denote $q^{(j)}$ as the ingress queue for the j^{th} flow, and all the queues other than ingress ones are called internal queues. Notice that $q^{(j)}$ and $q_\ell^{(j)}$ represent the same queue if link ℓ is the first link traversed by flow j in the network. The queues traversed right before and after queue q are named the upstream queue of q and the downstream queue(s) of q, respectively. For the clarity of our presentation, we assume at this moment that all the queues are of infinite sizes (including both ingress queues and internal queues) so that no packets are lost. Later, we will show that all internal queues are of finite sizes.

Throughout this paper, we consider the usual discrete-time setting by assuming that packets are of the same size and that time is slotted so that one packet can be transmitted within a time slot. Also, we assume that each queue is started from an empty system at time 0. We now define $x_\ell^{(j)}(t)$ as the number of packets in queue $q_\ell^{(j)}$ at time t. Then, the governing equation for each queue $q_\ell^{(j)}$ can be represented as

$$x_\ell^{(j)}(t+1) = x_\ell^{(j)}(t) + a_\ell^{(j)}(t+1) - b_\ell^{(j)}(t+1), \qquad (1)$$

where $a_\ell^{(j)}(t)$ and $b_\ell^{(j)}(t)$ are the number of packets arriving at $q_\ell^{(j)}$ at time t and the number of packets departing for the downstream queue(s) of $q_\ell^{(j)}$ or leaving the network at time t, respectively. Notice that if $q_\ell^{(j)}$ is the ingress queue for flow j (i.e., $q_\ell^{(j)} = q^{(j)}$), $a_\ell^{(j)}(t)$ is simply the number of external arrival packets at time t, which is denoted as $a^{(j)}(t)$. On the other hand, for internal queue $q_\ell^{(j)}$, the arrival packets of $q_\ell^{(j)}$ are exactly the departure packets of $q_{u(j,\ell)}^{(j)}$, where $u(j,\ell)$ is the upstream queue of link ℓ for flow j. That is,

$$a_\ell^{(j)}(t) = b_{u(j,\ell)}^{(j)}(t)$$

for an internal queue $q_\ell^{(j)}$.

In this paper, $x_\ell^{(j)}(t)$, $a_\ell^{(j)}(t)$ and $b_\ell^{(j)}(t)$ are nonnegative integers for flows and links. Moreover, we assume that each multicast traffic flow does not traverse any link twice. According to (1), all the downstream queues of $q_\ell^{(j)}$ receive the packets sent by $q_\ell^{(j)}$ in the same time slot. Such a multicast policy is named *no fanout splitting* in the literature [9].

B. Constrained simultaneous transmissions for links

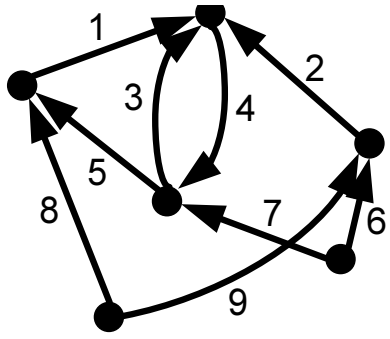

Fig. 1. A directed graph with 6 nodes and 9 directed links.

As there is interference in wireless networks, not every link can transmit at the same time. We model this by using *configuration vectors*. An L-vector $P = [p_1, p_2, \ldots, p_L]$ is called a *configuration vector* if, for all $1 \le \ell \le L$, exactly p_ℓ

packets can be transmitted through link ℓ in a time slot. Here we assume that p_ℓ's are either 1 or 0, and we say that $\ell \in P$ if $p_\ell = 1$ and $\ell \notin P$ otherwise.

Let W be the collection of all configuration vectors. In practice, W reflects both physical and topological constraints on the network. For example, consider the network represented by the directed graph in Figure 1, where each directed link represents that packets can be sent from the head of the link to the tail of the link. Under the node exclusive interference model [12], links 1 and 7 in Figure 1 can transmit packets at the same time. However, since each pair of links are separated by at most one link in Figure 1, there is at most one input/output pair can transmit a packet at the same time under the assumption of IEEE 802.11 based interference model [12]. As configuration vectors are mainly due to interference in wireless networks, it is reasonable to assume in this paper that the set W is coordinate convex, i.e., if $P_1 \leq P_2$ and P_2 is in W, then P_1 is also in W. Note that the inequality $P_1 \leq P_2$ holds *componentwise*, i.e., every component in P_1 is not greater than the corresponding component in P_2.

With the collection of configuration vectors W, the capacity region of the network, denoted by Γ, is known to be the convex hull of all the configuration vectors in W, namely,

$$\begin{aligned}\Gamma \;=\; & \{\mathbf{r} = (r_1, r_2, \ldots, r_L) | \exists\, \{\phi_k\}_{k=1}^K \text{ and } \\ & \{P_k\}_{k=1}^K \subset W \text{ such that } \phi_k \geq 0,\ \forall 1 \leq k \leq K, \\ & \sum_{k=1}^K \phi_k = 1 \text{ and } \mathbf{r} \leq \sum_{k=1}^K \phi_k P_k\}. \end{aligned} \tag{2}$$

Clearly, if $\sum_{k=1}^K \phi_k \leq 1$ for a set of nonnegative numbers $\{\phi_k\}_{k=1}^K$, then $\sum_{k=1}^K \phi_k P_k$ is in capacity region Γ. The set of nonnegative numbers $\{\phi_k\}_{k=1}^K$ are called the *weights* with respect to the configuration vectors $\{P_k\}_{k=1}^K$.

C. Routing Vectors and admissible traffic

In this paper, we only consider a fixed route for each multicast flow. For this, we define the L-vector $R^{(j)} = [R_1^{(j)}, R_2^{(j)}, \ldots, R_L^{(j)}]$ as the *routing vector* for flow j, where $R_\ell^{(j)} = 1$ if the j^{th} flow traverses link ℓ and $R_\ell^{(j)} = 0$ otherwise. Moreover, the set F_ℓ is used to denote all the flows traversing link ℓ, namely,

$$F_\ell = \{j | R_\ell^{(j)} = 1,\ 1 \leq j \leq J\}, \tag{3}$$

for all $1 \leq \ell \leq L$. For example, assume that there are two traffic flows in Figure 1. The first flow traverses through links 6, 2, 4 and 5 sequentially, and the second flow traverses through links 8, 1, and 4 sequentially. Thus, the routing vectors for the first and second flows in Figure 1 can be represented as $R^{(1)} = (0,1,0,1,1,1,0,0,0)$ and $R^{(2)} = (1,0,0,1,0,0,0,1,0)$, respectively. Also, according to (3), we have that $F_1 = \{2\}$ and $F_4 = \{1,2\}$.

Let $\lambda^{(j)}$ be the average number of flow j packets that arrive at the network in a time slot. For the ease of our presentation, we simply call $\lambda^{(j)}$ the arrival rate of flow j. With the routing vector for every flow in the network, we can then compute the average number of packets that need to go through a particular link in a time slot. Specifically, we have

$$\Lambda = (\lambda_1, \lambda_2, \ldots, \lambda_L) = \sum_{j=1}^J \lambda^{(j)} R^{(j)}, \tag{4}$$

where λ_ℓ, $\ell = 1, 2, \ldots, L$, is the average number of packets that need to go through link ℓ in a time slot. The vector Λ is called the arrival rate vector in this paper. In the following, we define the traffic intensity and admissible traffic.

Definition 1 *(***Intensity and admissible traffic***) The intensity* $\rho = ||\Lambda||$ *of* Λ *is defined as*

$$\begin{aligned} ||\Lambda|| = \inf_{\rho'} \{\rho' = \sum_{k=1}^K \phi_k | \exists \textit{ weights } \{\phi_k\}_{k=1}^K \textit{ and } \\ \{P_k\}_{k=1}^K \subset W \textit{ such that } \Lambda \leq \sum_{k=1}^K \phi_k P_k \in \Gamma\}. \end{aligned} \tag{5}$$

The input traffic is said to be admissible *if* $\rho < 1$.

If the input traffic is admissible, one can always find a set of weights $\{\phi_k\}_{k=1}^K$ and a set of configuration vectors $\{P_k\}_{k=1}^K$ such that $\sum_{k=1}^K \phi_k = 1$ and $\Lambda < \sum_{k=1}^K \phi_k P_k$. In view of this, a simple time sharing policy that schedules configuration vector P_k proportional to the weight ϕ_k guarantees that the arrival rate at each link is strictly smaller than the service rate of that link. If, furthermore, each flow is stationary and ergodic with a *known* rate, then there are many scheduling polices in the literature that can be used for stabilizing each queue in the network (see e.g., the rate proportional processor sharing (RPPS) scheme in [10] and the service curve earliest deadline first (SCED) scheme in [11]). However, if the arrival rates are *not known*, then most packet scheduling polices that stabilize any admissible traffic in the capacity region are related to the maximum weighted matching (MWM) algorithm in [13], where the most suitable configuration vector is identified in every time slot. In the next section, we will extend the dynamic frame sizing algorithm for switches and *wired* networks in [3], [8] to the mathematical model for wireless networks described in this paper.

III. Dynamic Frame Sizing Algorithm

In [3], [8], the dynamic frame sizing (DFS) algorithm has been used for stabilizing queues in switches and wired networks without knowing the arrival rates. In the DFS algorithm, time is partitioned into frames, where the frame size is not fixed and is determined at the beginning of each frame. The main idea of the DFS algorithm is to determine the *minimum* frame size at the beginning of a frame so that the backlog observed at the ingress queue of each flow at the beginning of the frame can be cleared by the end of the frame. To ensure the number of packets of each flow inside the network is bounded above by a finite constant, the DFS algorithm then provides each flow in a frame a guaranteed rate that is proportional to the backlog observed at the ingress queue at the beginning

of that frame. As long as the expected size of each frame is finite, the expected backlog at each queue remains finite.

In view of this, there are two steps for the extension of the DFS algorithm to the mathematical model for wireless networks described in Section II.

(i) Determine the frame size at the beginning of each frame.

(ii) Provide guaranteed rate services in our mathematical model.

We will address the first step in Section III-A. By proposing a hierarchical smooth algorithm in Section III-B, we show how to provide guaranteed rate services in our mathematical model. The proof for the finiteness for the expected frame size under the DFS algorithm will be given in Section III-C.

A. Determining the frame size

In this section, we present the method for determining the frame size at the beginning of each frame in the DFS algorithm.

(S1) Denote by τ_n the last time slot of the $(n-1)^{th}$ frame (and by τ_n+1 the beginning time slot of the n^{th} frame). Suppose that there are $x^{(j)}(\tau_n)$ packets stored in the ingress queue $q^{(j)}$ for flow j at the beginning of the n^{th} frame, $j=1,2,\ldots,J$. The *workload* of the n^{th} frame $Y(n)=(y_1(n),y_2(n),\ldots,y_L(n))$ can be thus defined as

$$Y(n)=\sum_{j=1}^{J} x^{(j)}(\tau_n)R^{(j)}, \tag{6}$$

where $R^{(j)}$ is the routing vector of flow j. That is, $y_\ell(n)$ is the total number of packets that need to be sent through link ℓ in the n^{th} frame. If $y_\ell(n)=0$ for all $\ell=1,2,\ldots,L$, then we simply set the size of the n^{th} frame, denoted by T_n, to be 1. Otherwise, T_n is defined as follows:

$$\begin{aligned} T_n &= \inf_{T'}\{T'=\sum_{k=1}^{K} m_k | \exists \text{ positive integers} \\ &\quad \{m_k\}_{k=1}^{K} \text{ and } \{P_k\}_{k=1}^{K}\subset W \\ &\quad \text{such that } \sum_{k=1}^{K} m_k P_k \geq Y(n)\}. \end{aligned} \tag{7}$$

As the set of configuration vectors is coordinate convex, one can always replace a configuration vector by a smaller configuration vector in the minimization problem in (7). As such, the minimization problem in (7) can be written equivalently (with an equality and possibly a larger set of configuration vectors) as follows:

$$\begin{aligned} T_n &= \inf_{T'}\{T'=\sum_{k=1}^{K} m_k | \exists \text{ positive integers} \\ &\quad \{m_k\}_{k=1}^{K} \text{ and } \{P_k\}_{k=1}^{K}\subset W \\ &\quad \text{such that } \sum_{k=1}^{K} m_k P_k = Y(n)\}. \end{aligned} \tag{8}$$

To see the intuition of T_n, suppose that $\sum_{k=1}^{K} m_k P_k = Y(n)$ for some m_k and P_k, $k=1,2,\ldots,K$. If we set the configuration vector P_k for m_k times for all $k=1,2,\ldots,K$, then the workload at every link will be cleared. As T_n is the minimum amount of time to do that, it is known as the *minimum clearance time*. Note that the problem of determining the minimum clearance time in switches and wired networks are quite simple and in fact there are explicit formulae as shown in [3], [8]. However, the problem here is much more difficult as it is formulated as an integer programming problem.

Without an explicit representation for the frame size T_n, we cannot simply follow the proofs in [3], [8] to show the finiteness of the expected frame size. One of the main contributions of this paper is the following upper bound for the frame size T_n that allows us to extend the result for wired networks to our mathematical model for wireless networks here.

The idea for the upper bound is to consider the smooth schedule in [6]. For a set of configuration vectors $\{P_k\}_{k=1}^{K}$ and their weights $\{\phi_k\}_{k=1}^{K}$, the smooth schedule in [6] assigns each configuration vector a class of tokens. The n^{th} token of configuration vector P_k is assigned with the eligible time $1+\lfloor (n-1)/\phi_k \rfloor$ and the deadline $\lceil n/\phi_k \rceil$. That is, there is a token generated and eliminated every $1/\phi_k$ time slots for configuration vector P_k. At each time slot t, the schedule selects one eligible token with the earliest deadline (the EDF policy in the literature) among all the remaining tokens and removes that token. If the token for configuration vector P_k is selected at the beginning of time slot t, then each link $\ell \in P_k$ is allowed to send a packet in that time slot. It is shown in Lemma 3.1. in [8] that each token is selected not later than its deadline in such a smooth schedule if the sum of the weights is not greater than 1, i.e., $\sum_{k=1}^{K}\phi_k \leq 1$.

Lemma 2 *For a set of configuration vectors $\{P_k\}_{k=1}^{K}$ and their weights $\{\phi_k\}_{k=1}^{K}$, let S_ℓ be the set of the configuration vectors containing link ℓ, namely,*

$$S_\ell=\{k | \textit{the } \ell^{th} \textit{ element of } P_k>0, 1\leq k\leq K\}, \tag{9}$$

and ψ_ℓ be the aggregate weight for link ℓ, namely,

$$\psi_\ell=\sum_{k\in S_\ell}\phi_k.$$

If $\sum_{k=1}^{K}\phi_k \leq 1$ and $\psi_\ell>0$ for each link ℓ with $y_\ell(n)>0$, then

$$T_n \leq \max_{1\leq \ell\leq L}\left\lceil \frac{y_\ell(n)+|S_\ell|}{\psi_\ell}\right\rceil + 1. \tag{10}$$

Proof. If $y_\ell(n)=0$ for all $\ell=1,2,\ldots,L$, then we know $T_n=1$ and the bound in (10) holds trivially. Now assume there exist some ℓ such that $y_\ell(n)>0$. For this case, T_n is the minimum clearance time defined in (7). As such, T_n is upper bounded by the time needed to clear all the backlogs by using the smooth schedule with the set of configuration vectors $\{P_k\}_{k=1}^{K}$ and their weights $\{\phi_k\}_{k=1}^{K}$. Since each token is selected not later than its deadline in such a smooth schedule,

the total number of tokens that are selected for configuration vector P_k by time t is at least $\lfloor \phi_k t \rfloor$ (as the deadline of the $\lfloor \phi_k t \rfloor^{th}$ token for configuration vector P_k is $\lceil \lfloor \phi_k t \rfloor / \phi_k \rceil \leq t$). This then implies that the total number of packets that link ℓ can transmit by time t is at least $\sum_{k \in S_\ell} \lfloor \phi_k t \rfloor$. Let

$$T = \max_{1 \leq \ell \leq L} \left\lceil \frac{y_\ell(n) + |S_\ell|}{\psi_\ell} \right\rceil.$$

Then, the total number of packets sent out from link ℓ in the duration $[0, T]$ is at least

$$\begin{aligned} \sum_{k \in S_\ell} \lfloor \phi_k T \rfloor &\geq \sum_{k \in S_\ell} (\phi_k T - 1) \\ &\geq \sum_{k \in S_\ell} \phi_k \left(\frac{y_\ell(n) + |S_\ell|}{\psi_\ell}\right) - |S_\ell| = y_\ell(n) \end{aligned} \quad (11)$$

for all $1 \leq \ell \leq L$. Thus, by time T, the backlog in every link is cleared by using the smooth schedule. Clearly, we have

$$T \leq \max_{1 \leq \ell \leq L} \left[\frac{y_\ell(n) + |S_\ell|}{\psi_\ell} \right] + 1,$$

and the upper bound also holds for this case. ■

B. A hierarchical smooth schedule

In this section, we propose a hierarchical smooth schedule that is able to provide guaranteed rate services in our mathematical model for wireless networks. There are two levels of this schedule: (i) the upper level for configuration vectors and (ii) the lower level for flows in each link.

(S2) The smooth schedule for configuration vectors:

Suppose the next frame size T_n is determined in the minimization problem in (8) with the set of configuration vectors $\{P_k\}_{k=1}^K$ and the set $\{m_k\}_{k=1}^K$ (the index n is omitted here for clarity). During the n^{th} frame, run the smooth schedule with weights m_k/T_n. As $T_n = \sum_{k=1}^K m_k$ in the minimization problem in (8), the sum of the weights is 1 in this smooth schedule, i.e., $\sum_{k=1}^K m_k/T_n = 1$. Specifically, the i^{th} token of the configuration vector P_k is assigned with the eligible time $\tau_n + 1 + \lfloor (i-1)T_n/m_k \rfloor$ and deadline $\tau_n + \lceil iT_n/m_k \rceil$, $1 \leq i \leq m_k$, $1 \leq k \leq K$. Then, for each time slot t in the interval $[\tau_n + 1, \tau_n + T_n]$, the smooth schedule selects the eligible token with the earliest deadline, and assigns the corresponding configuration vector at time t. At the beginning of the n^{th} frame, the schedule for all the configuration vectors in $[\tau_n + 1, \tau_n + T_n]$ is computed and then transmitted to all the links in the network.

(S3) The smooth schedule for flows in each link:

Since $\{m_k\}_{k=1}^K$ and $\{P_k\}_{k=1}^K$ achieve the minimization problem in (8), we have that

$$\sum_{k=1}^K m_k P_k = Y(n). \quad (12)$$

Similar to (9), let S_ℓ be the set of the configuration vectors in $\{P_k\}_{k=1}^K$ containing link ℓ within the n^{th} frame. Thus, we have from (12) and (6) that

$$\sum_{k \in S_\ell} m_k = y_\ell(n) = \sum_{j \in F_\ell} x^{(j)}(\tau_n), \quad (13)$$

where F_ℓ, $1 \leq \ell \leq L$, is the set of all traffic flows traversing link ℓ as defined in (3). We now say that a link is allowed to transmit a packet or simply *allowable* at time t if a configuration vector containing the link is selected at that time slot. Then, according to (13), there are totally $y_\ell(n)$ allowable time slots for link ℓ in the n^{th} frame. For these $y_\ell(n)$ allowable time slots, we denote $H_\ell(i)$ as the i^{th} allowable time slot for link ℓ in this frame. The lower level schedule for link ℓ now generates $x^{(j)}(\tau_n)$ tokens for each flow j in F_ℓ within the n^{th} frame. Then, in the n^{th} frame, the i^{th} token of flow j is assigned with the eligible time $H_\ell\left(1 + \left\lfloor (i-1)y_\ell(n)/x^{(j)}(\tau_n) \right\rfloor\right)$ and the deadline $H_\ell\left(\left\lceil i \cdot y_\ell(n)/x^{(j)}(\tau_n) \right\rceil\right)$. When the link ℓ is allowed to transmit one packet, namely, the upper schedule for configuration vectors selects some P_k that contains link ℓ, the lower level schedule for link ℓ selects the eligible token that has the earliest deadline among all the flows in F_ℓ and removes that token. Suppose the selected token is for flow j, then queue $q_\ell^{(j)}$ is allowed to send a packet in that time slot. In other words, the lower level schedule is also a smooth schedule for the traffic flows traversing link ℓ with the weights $\{x^{(j)}(\tau_n)/y_\ell(n)\}_{j \in F_\ell}$ on all the $y_\ell(n)$ allowable time slots for link ℓ. Note from (13) that the sum of the weights is also 1, i.e., $\sum_{j \in F_\ell} x^{(j)}(\tau_n)/y_\ell(n) = 1$.

As both the sums of the weights in (S2) and (S3) are 1, the following lemma is a direct consequence of the well-known result for a smooth schedule (see e.g., Lemma 3.1 in [8]).

Lemma 3 *Under the hierarchical smooth schedule described in (S2) and (S3),*

(i) *each token in the smooth schedule for the configuration vectors is selected not later than its deadline, and*
(ii) *each token in the smooth schedule for flows in each link is also selected not later than its deadline.*

In the following lemma, we derive an upper bound and a lower bound on the total number of tokens selected during an interval for a particular flow by a link traversed by that flow. These bounds will be used to bound the total number of packets in an internal buffer for each flow in Theorem 5. The proof of Lemma 4 is given in Appendix A.

Lemma 4 *Let $C_\ell^{(j)}(t)$ be the cumulative number of tokens selected for flow j by link ℓ by time t. Suppose s is a time slot in the ${n_1}^{th}$ frame and $t > s$ is another time slot in the ${n_2}^{th}$ frame. Then, for each link ℓ traversed by flow j, the total number of tokens selected for flow j by link ℓ in the time interval $[s+1, t]$, i.e., $C_\ell^{(j)}(t) - C_\ell^{(j)}(s)$, have the following upper and lower bounds:*

$$\begin{aligned} &E^{(j)}(s,t) - 2(S_{\max} + 1) \leq C_\ell^{(j)}(t) - C_\ell^{(j)}(s) \\ &\leq E^{(j)}(s,t) + 2(S_{\max} + 1), \end{aligned} \quad (14)$$

where

$$E^{(j)}(s,t) = \sum_{n'=n_2}^{n_1-1} x^{(j)}(\tau_{n'}) + \left(x^{(j)}(\tau_{n_2}) \frac{t-\tau_{n_2}}{T_{n_2}} - x^{(j)}(\tau_{n_1}) \frac{s-\tau_{n_1}}{T_{n_1}} \right), \quad (15)$$

and $S_{\max}$ is the maximum size of the set of all the configuration vectors in W containing link ℓ, namely,

$$S_{\max} = \max_{1 \le \ell \le L} |\{P | \ell \in P, P \in W\}|. \quad (16)$$

Now we show in the following theorem that the total number of packets in an internal buffer is bounded by a finite constant.

Theorem 5 *Under the DFS algorithm specified in (S1), (S2) and (S3), the total number of packets of a flow in any internal queue traversed by the flow is upper bounded by $4(S_{\max}+1)$ for any time slot, where $S_{\max}$ is the constant defined in (16). Specifically, suppose that $q_\ell^{(j)}$ is an internal queue for flow j at link ℓ. Then*

$$x_\ell^{(j)}(t) \le 4(S_{\max}+1), \quad (17)$$

where $x_\ell^{(j)}(t)$ is the number of packets in $q_\ell^{(j)}$ at time t.

Proof. Consider the governing equation of queue $q_\ell^{(j)}$ in (1). Note that there is a departure from $q_\ell^{(j)}$ at time t, i.e., $b_\ell^{(j)}(t) = 1$, if a token is selected for flow j by link ℓ at time t and there are packets that can be departed from the queue. Let $c_\ell^{(j)}(t)$ be the indicator variable for the event that a token is selected for flow j by link ℓ at time t. Thus, we can rewrite the governing equation of queue $q_\ell^{(j)}$ in (1) as follows:

$$x_\ell^{(j)}(t+1) = \max[0, x_\ell^{(j)}(t) + a_\ell^{(j)}(t+1) - c_\ell^{(j)}(t+1)]. \quad (18)$$

As we assume the queue is empty at time 0, recursively expanding the governing equation in (18) yields

$$x_\ell^{(j)}(t) = \max_{0 \le s \le t} \left[A_\ell^{(j)}(t) - A_\ell^{(j)}(s) - \left(C_\ell^{(j)}(t) - C_\ell^{(j)}(s) \right) \right], \quad (19)$$

where $A_\ell^{(j)}(t) = \sum_{t_1=1}^{t} a_\ell^{(j)}(t_1)$ and $C_\ell^{(j)}(t) = \sum_{t_1=1}^{t} c_\ell^{(j)}(t_1)$ are the accumulative number of packets arriving at $q_\ell^{(j)}$ and the accumulative number of tokens selected for flow j by link ℓ by time t, respectively. Recall that $u(j,\ell)$ is the upstream link of link ℓ for flow j. For link ℓ, we define $B_\ell^{(j)}(t) = \sum_{t_1=1}^{t} b_\ell^{(j)}(t_1)$ as the cumulative number of packets that depart from $q_\ell^{(j)}$ by time t. Since a packet can depart from an upstream queue only if there is a token selected for the upstream queue, it then follow that

$$A_\ell^{(j)}(t) - A_\ell^{(j)}(s) = B_{u(j,\ell)}^{(j)}(t) - B_{u(j,\ell)}^{(j)}(s) \le C_{u(j,\ell)}^{(j)}(t) - C_{u(j,\ell)}^{(j)}(s), \quad (20)$$

for each internal queue $q_\ell^{(j)}$.

From (20) and Lemma 4, it follows that

$$A_\ell^{(j)}(t) - A_\ell^{(j)}(s) \le C_{u(j,\ell)}^{(j)}(t) - C_{u(j,\ell)}^{(j)}(s) \le E^{(j)}(s,t) + 2(S_{\max}+1). \quad (21)$$

According to Lemma 4, we have that

$$C_\ell^{(j)}(t) - C_\ell^{(j)}(s) \ge E^{(j)}(s,t) - 2(S_{\max}+1). \quad (22)$$

Thus, the result in (17) follows directly from (19), (21) and (22). ■

C. Frame Bound

In this section, we show that, for Bernoulli arrival traffic with arrival rates inside the capacity region, the expected frame size is finite. Thus, the DFS algorithm guarantees 100% throughput for such Bernoulli traffic.

Here we make three specific assumptions on the input traffic.

(A1) All the multicast flows are independent Bernoulli processes when they arrive at the network. Specifically, $a^{(j)}(t)$'s are independent Bernoulli random variables for all j and t.

(A2) Assume that the arrival rate of flow j is $\lambda^{(j)}$, $1 \le j \le J$, and this rate information is unknown to the network. Without loss of generality, we also assume $\lambda^{(j)} > 0$ for all j and every link is traversed by at least one flow, i.e.,

$$\mathbf{0} < \Lambda = (\lambda_1, \lambda_2, \ldots, \lambda_L) = \sum_{j=1}^{J} \lambda^{(j)} R^{(j)},$$

where $\mathbf{0}$ is the L-vector with all its elements being 0.

(A3) The input traffic is admissible, i.e., the intensity defined in (5) is strictly smaller than 1, i.e., $\rho = ||\Lambda|| < 1$.

As we assume that the traffic is admissible, there is a set of weights $\{\phi_k\}_{k=1}^K$ and a set of configuration vectors $\{P_k\}_{k=1}^K$ such that

$$0 < \rho = \sum_{k=1}^{K} \phi_k < 1, \quad (23)$$

and

$$\mathbf{0} < \Lambda \le \sum_{k=1}^{K} \phi_k P_k. \quad (24)$$

Theorem 6 *Assume that the input traffic satisfies (A1)–(A3). Consider the set of weights $\{\phi_k\}_{k=1}^K$ and the set of configuration vectors $\{P_k\}_{k=1}^K$ that satisfy (23) and (24). Let $\psi_\ell = \sum_{k \in S_\ell} \phi_k / \rho$, where S_ℓ is the set of configuration vectors containing link ℓ in $\{P_k\}_{k=1}^K$ as in (9). Also, let $\psi_{\min} = \min_{1 \le \ell \le L} \psi_\ell$ and $\delta = \max_{1 \le \ell \le L} (|S_\ell|/\psi_\ell) + 1$. Then, under the DFS algorithm specified in (S1), we have for $n > 1$*

$$\log E[e^{\theta^* T_n}] \le \max\left(\theta^*, \frac{2\log L + 2\delta\theta^*}{1-\rho} \right) \quad (25)$$

where θ^* is the unique positive solution of

$$\frac{e^{\theta/\psi_{\min}} - 1}{\theta/\psi_{\min}} = \frac{1+\rho}{2\rho}. \quad (26)$$

As a results, the expectation of the frame size $E[T_n]$ is bounded by $\max\left(\frac{2\log L+2\delta\theta^*}{\theta^*(1-\rho)}\right)$.

As a direct consequence of Theorem 6, the expected number of packets in each ingress queue is also finite. In conjunction with the bound for an internal queue in Theorem 5, the DFS algorithm stabilizes the network for the Bernoulli traffic described in (A1)–(A3).

Proof.

Let $\tilde{\phi}_k = \phi_k/\rho$. It then follows from (23) that

$$\sum_{k=1}^{K} \tilde{\phi}_k = 1. \tag{27}$$

Moreover, we have from (24) that

$$\psi_\ell = \sum_{k\in S_\ell} \tilde{\phi}_k = \sum_{k\in S_\ell} \frac{\phi_k}{\rho} \geq \frac{\lambda_\ell}{\rho} > 0. \tag{28}$$

Thus, the condition in Lemma 2 is satisfied (with the set of weights $\{\tilde{\phi}_k\}_{k=1}^K$ and the set of configuration vectors $\{P_k\}_{k=1}^K$) and we have from (10) that

$$T_{n+1} \leq \max_{1\leq\ell\leq L}\left[\frac{y_\ell(n+1)+|S_\ell|}{\psi_\ell}\right]+1. \tag{29}$$

For $\theta > 0$, we have that

$$\begin{aligned} e^{\theta T_{n+1}} &\leq \max_{1\leq\ell\leq L} \exp\left[\theta\left(\frac{y_\ell(n+1)+|S_\ell|}{\psi_\ell}+1\right)\right] \\ &\leq \sum_{1\leq\ell\leq L} \exp\left[\theta\left(\frac{y_\ell(n+1)+|S_\ell|}{\psi_\ell}+1\right)\right]. \end{aligned}$$

According to (6) and (13), the workload $y_\ell(n+1)$ for link ℓ can be represented by

$$y_\ell(n+1) = \sum_{j\in F_\ell} x^{(j)}(\tau_{n+1}) = \sum_{j\in F_\ell}\sum_{t=\tau_n+1}^{\tau_{n+1}} a^{(j)}(t), \tag{30}$$

where $a^{(j)}(t)$ is the external arrival packets for flow j at time t and F_ℓ is the set that contains all the traffic flows traversing link ℓ as described in (3). Since we assume that the arrival processes are independent Bernoulli processes in (A1) and $\{a^{(j)}(t)\}_{t=1}^{\infty}$ are i.i.d Bernoulli random variables with parameter $\lambda^{(j)}$ in (A2), it then follows that

$$\begin{aligned} E[e^{\theta T_{n+1}}|T_n] \leq \sum_{1\leq\ell\leq L} & e^{\theta(1+|S_\ell|/\psi_\ell)} \\ & \left\{E\left[\exp\left(\frac{\theta\sum_{j\in F_\ell} a^{(j)}(1)}{\psi_\ell}\right)\right]\right\}^{T_n}. \end{aligned} \tag{31}$$

Notice that

$$\begin{aligned} & \log E\left[\exp\left(\frac{\theta\sum_{j\in F_\ell} a^{(j)}(1)}{\psi_\ell}\right)\right] \\ = & \sum_{j\in F_\ell} \log E\left[\exp\left(\frac{\theta a^{(j)}(1)}{\psi_\ell}\right)\right] \\ = & \sum_{j\in F_\ell} \log\left(\lambda^{(j)}e^{\theta/\psi_\ell}+1-\lambda^{(j)}\right) \\ \leq & \sum_{j\in F_\ell} \lambda^{(j)}(e^{\theta/\psi_\ell}-1) \\ = & \lambda_\ell(e^{\theta/\psi_\ell}-1) \\ \leq & \rho\psi_\ell(e^{\theta/\psi_\ell}-1), \end{aligned} \tag{32}$$

where we use $\log(1+x)\leq x$ for $x>0$ in the first inequality and (28) in the last inequality.

Hence, we have that

$$E\left[e^{\theta T_{n+1}}|T_n\right] \leq e^{\theta\delta}\sum_{1\leq\ell\leq L}\exp\left[\rho T_n\psi_\ell(e^{\theta/\psi_\ell}-1)\right]. \tag{33}$$

For fixed $\theta>0$, according to Taylor's expansion, we have that

$$\begin{aligned} \psi_\ell(e^{\theta/\psi_\ell}-1) &= \psi_\ell\left[\sum_{k=0}^{\infty}\frac{1}{k!}\left(\frac{\theta}{\psi_\ell}\right)^k-1\right] \\ &= \psi_\ell\left(\frac{\theta}{\psi_\ell}+\frac{1}{2!}\frac{\theta^2}{\psi_\ell^2}+\frac{1}{3!}\frac{\theta^3}{\psi_\ell^3}\cdots\right) \\ &= \theta+\sum_{k=1}^{\infty}\frac{\theta^{k+1}}{(k+1)!\psi_\ell^k}, \end{aligned} \tag{34}$$

which decreases monotonically as ψ_ℓ increases. As such,

$$\psi_\ell(e^{\theta/\psi_\ell}-1) \leq \psi_{\min}(e^{\theta/\psi_{\min}}-1) \tag{35}$$

for all $1\leq\ell\leq L$. From (33) and (35), we have that

$$E\left[e^{\theta T_{n+1}}|T_n\right] \leq e^{\theta\delta}L\exp\left(\rho T_n\psi_{\min}(e^{\theta/\psi_{\min}}-1)\right). \tag{36}$$

Taking expectation on both sides of (36) yields

$$E[e^{\theta T_{n+1}}] \leq e^{\theta\delta}LE\left[\exp(\rho T_n\psi_{\min}(e^{\theta/\psi_{\min}}-1))\right]. \tag{37}$$

According to (26), we have that

$$\rho\psi_{\min}(e^{\theta^*/\psi_{\min}}-1) = \theta^*(\rho+1)/2,$$

and (37) can be rewritten (with θ being replaced by θ^*) as

$$E[e^{\theta^* T_{n+1}}] \leq e^{\theta^*\delta}LE[e^{\theta^* T_n(1+\rho)/2}]. \tag{38}$$

Since $\log E[e^{\theta T_n}]$ is convex in θ (see e.g., [1, Proposition 7.1.8]) and $\rho<1$, we have that

$$\log E[e^{\theta^* T_n(1+\rho)/2}] \leq \frac{1+\rho}{2}\log E[e^{\theta^* T_n}]. \tag{39}$$

Using (39) and (38) yields

$$\log E[e^{\theta^* T_{n+1}}] \leq \log L+\delta\theta^*+\frac{1+\rho}{2}\log E[e^{\theta^* T_n}]. \tag{40}$$

Since $T_1=1$ (as the network is started from an empty system), one can verify (25) from induction by using (40). Finally, we

use (25) to show the bound of the frame size in Theorem 6. Since $e^{\theta x}$ is convex in x, it follows from Jensen's Inequality that

$$E[T_n] \leq \frac{1}{\theta^*} \log E[e^{\theta^* T_n}] \leq \max\left(1, \frac{2\log L + 2\delta\theta^*}{\theta^*(1-\rho)}\right). \quad (41)$$

■

IV. Conclusion

In this paper, we extended the dynamic frame sizing algorithm to the setting of wireless networks. We modeled wireless networks by configuration vectors that specify the sets of links that can transmit at the same time. For such a mathematical model, we considered multicast flows with per flow queueing. We proved that the DFS algorithm indeed stabilizes the network for any admissible Bernoulli traffic. In comparison with the previous results for the DFS algorithm in [3], [8], our main contributions in this paper are the two new technical results: (i) an upper bound for the frame size that has an explicit expression in terms of the workload, and (ii) a hierarchical smooth schedule that provides guaranteed rate services in such a mathematical model. The first result allows us to modify the large deviation argument in [3], [8] to prove the finiteness of the expected frame size, while the second result leads to an upper bound for the total number of packets in an internal queue.

There are some possible extensions of this work.

(i) In (S1), the frame size is chosen to be 1 when there is no backlog in each ingress queue. In the worst case, this requires the optimization problem to be carried out in every time slot (which is as bad as the maximum weighted matching algorithm). We note that it is possible to set a bound on the minimum frame size so that the optimization problem needs not be carried out too often. This is because our proof only relies on the assumption that the backlog in each ingress queue at the beginning of a frame needs to be cleared by the end of that frame.

(ii) The DFS algorithm described in this paper does not provide traffic isolation. When the traffic is not admissible, the expected frame size cannot be bounded. As all the flows are coupled through the optimization problem that determines the frame size at the beginning of each frame, the performance could be very bad for all the flows. In view of this, one should enforce an upper limit for the frame size. But this also limits the throughput that can be achieved by the DFS algorithm.

V. Appendix A

In the proof of Lemma 4, we need to consider multiple frames. As such, we add the index n in $K(n)$, $m_k(n)$, $P_k(n)$, $S_\ell(n)$, and $F_\ell(n)$ to represent all the corresponding parameters used in the DFS algorithm within frame n. First, we consider a time slot t in the n^{th} frame. Define $D_\ell(t)$ as the cumulative number of allowable time slots for link ℓ by time t. According to (S2) and (S3), when a token for the configuration vector $P_k(n) \in S_\ell(n)$ is selected by the schedule at a time slot, then link ℓ is allowable at that time slot. Prior to the n^{th} frame, we know from (13) that there are totally $\sum_{n'=1}^{n-1} y_\ell(n')$ tokens selected for link ℓ.

Now, consider the n^{th} frame that t belongs to. First, the cumulative number of allowable time slots for link ℓ by time t in the n^{th} frame is $D_\ell(t) - \sum_{n'=1}^{n-1} y_\ell(n')$ and this number cannot be greater than the total number of tokens generated for configuration vectors containing link ℓ. On the other hand, from Lemma 3(i), all the tokens for configuration vectors $\{P_k(n)\}_{k=1}^{K(n)}$ are selected before their deadlines. As such, the allowable time slots for link ℓ in the n^{th} frame is not less than the total number of tokens for configuration vectors containing link ℓ with deadlines not later than t. Thus, for $\tau_n + 1 \leq t \leq \tau_n + T_n$, we have the following upper bound and lower bound:

$$\sum_{k\in S_\ell(n)} \left\lfloor \frac{t-\tau_n}{T_n} m_k(n) \right\rfloor \leq D_\ell(t) - \sum_{n'=1}^{n-1} y_\ell(n') \leq \sum_{k\in S_\ell(n)} \left\lceil \frac{t-\tau_n}{T_n} m_k(n) \right\rceil. \quad (42)$$

In the n^{th} frame, we also know from (13) that there are $x^{(j)}(\tau_n)$ tokens selected for flow j by link ℓ. Hence, before the n^{th} frame, there are $\sum_{n'=1}^{n-1} x^{(j)}(\tau_{n'})$ tokens selected by link ℓ for flow j. Within the n^{th} frame, the cumulative number of tokens *selected* for flow j by link ℓ by time t is upper bounded by the number of tokens *generated* for flow j by link ℓ by time t. Thus, it follows from the schedule in (S3) that

$$C_\ell^{(j)}(t) - \sum_{n'=1}^{n-1} x^{(j)}(\tau_{n'}) \leq \left\lceil \frac{x^{(j)}(\tau_n)}{y_\ell(n)} \left(D_\ell(t) - \sum_{n'=1}^{n-1} y_\ell(n') \right) \right\rceil. \quad (43)$$

On the other hand, from Lemma 3(ii), each token for flow j is selected before its deadline. As such, the cumulative number of tokens selected for flow j by link ℓ by time t is lower bounded by the tokens with deadlines not later than t. This then implies

$$C_\ell^{(j)}(t) - \sum_{n'=1}^{n-1} x^{(j)}(\tau_{n'}) \geq \left\lfloor \frac{x^{(j)}(\tau_n)}{y_\ell(n)} \left(D_\ell(t) - \sum_{n'=1}^{n-1} y_\ell(n') \right) \right\rfloor. \quad (44)$$

It follows from (43), (44) and (42) that

$$\left\lfloor \frac{x^{(j)}(\tau_n)}{y_\ell(n)} \sum_{k\in S_\ell(n)} \left\lfloor \frac{t-\tau_n}{T_n} m_k(n) \right\rfloor \right\rfloor \leq C_\ell^{(j)}(t) - \sum_{n'=1}^{n-1} x^{(j)}(\tau_{n'})$$

$$\leq \left\lceil \frac{x^{(j)}(\tau_n)}{y_\ell(n)} \sum_{k\in S_\ell(n)} \left\lceil \frac{t-\tau_n}{T_n} m_k(n) \right\rceil \right\rceil \quad (45)$$

for each time slot t in the n^{th} frame, namely, $\tau_n + 1 \leq t \leq \tau_n + T_n$.

Now suppose s is a time slot in the ${n_1}^{th}$ frame and $t > s$ is another time slot in the ${n_2}^{th}$ frame. Using (45) yields

$$\begin{aligned}
&C_\ell^{(j)}(t) - C_\ell^{(j)}(s) \\
&\geq \sum_{n'=n_1}^{n_2-1} x^{(j)}(\tau_{n'}) \\
&+ \left\lceil \frac{x^{(j)}(\tau_{n_2})}{y_\ell(n_2)} \sum_{k\in S_\ell(n_2)} \left\lfloor \frac{t-\tau_{n_2}}{T_{n_2}} m_k(n_2) \right\rfloor \right\rceil \\
&- \left\lceil \frac{x^{(j)}(\tau_{n_1})}{y_\ell(n_1)} \sum_{k\in S_\ell(n_1)} \left\lceil \frac{s-\tau_{n_1}}{T_{n_1}} m_k(n_1) \right\rceil \right\rceil \\
&\geq \sum_{n'=n_1}^{n_2-1} x^{(j)}(\tau_{n'}) \\
&+ \frac{x^{(j)}(\tau_{n_2})}{y_\ell(n_2)} \sum_{k\in S_\ell(n_2)} \left\lfloor \frac{t-\tau_{n_1}}{T_{n_2}} m_k(n_2) \right\rfloor \\
&- \frac{x^{(j)}(\tau_{n_1})}{y_\ell(n_1)} \sum_{k\in S_\ell(n_1)} \left\lceil \frac{s-\tau_{n_1}}{T_{n_1}} m_k(n_1) \right\rceil - 2 \\
&\geq \sum_{n'=n_1}^{n_2-1} x^{(j)}(\tau_{n'}) \\
&+ \frac{x^{(j)}(\tau_{n_2})}{y_\ell(n_2)} \left(\frac{t-\tau_{n_1}}{T_{n_2}} \sum_{k\in S_\ell(n_2)} m_k(n_2) - |S_\ell(n_2)| \right) \\
&- \frac{x^{(j)}(\tau_{n_1})}{y_\ell(n_1)} \left(\frac{s-\tau_{n_1}}{T_{n_1}} \sum_{k\in S_\ell(n_1)} m_k(n_1) + |S_\ell(n_1)| \right) - 2 \\
&= \sum_{n'=n_1}^{n_2-1} x^{(j)}(\tau_{n'}) \\
&+ \left(x^{(j)}(\tau_{n_2}) \frac{t-\tau_{n_2}}{T_{n_2}} - x^{(j)}(\tau_{n_1}) \frac{s-\tau_{n_1}}{T_{n_1}} \right) \\
&- \frac{x^{(j)}(\tau_{n_2})}{y_\ell(n_2)} |S_\ell(n_2)| - \frac{x^{(j)}(\tau_{n_1})}{y_\ell(n_1)} |S_\ell(n_1)| - 2, \quad (46)
\end{aligned}$$

where we use (13) in the last equality. Notice from (16) that $S_\ell(n) \leq S_{\max}$ for each frame n. Then, it follows from (46) that

$$\begin{aligned}
C_\ell^{(j)}(t) - C_\ell^{(j)}(s) &\geq \sum_{n'=n_1}^{n_2-1} x^{(j)}(\tau_{n'}) \\
&+ \left(x^{(j)}(\tau_{n_2}) \frac{t-\tau_{n_2}}{T_{n_2}} - x^{(j)}(\tau_{n_1}) \frac{s-\tau_{n_1}}{T_{n_1}} \right) \\
&- 2(S_{\max}+1). \quad (47)
\end{aligned}$$

Also, from (45), we have that

$$\begin{aligned}
C_\ell^{(j)}(t) - C_\ell^{(j)}(s) &\leq \sum_{n'=n_1}^{n_2-1} x^{(j)}(\tau_{n'}) \\
&+ \left\lceil \frac{x^{(j)}(\tau_{n_2})}{y_\ell(n_2)} \sum_{k\in S_\ell(n_2)} \left\lceil \frac{t-\tau_{n_1}}{T_{n_2}} m_k(n_2) \right\rceil \right\rceil \\
&- \left\lfloor \frac{x^{(j)}(\tau_{n_1})}{y_\ell(n_1)} \sum_{k\in S_\ell(n_1)} \left\lfloor \frac{s-\tau_{n_1}}{T_{n_1}} m_k(n_1) \right\rfloor \right\rfloor . \quad (48)
\end{aligned}$$

By following a similar procedure, one can verify that

$$\begin{aligned}
C_\ell^{(j)}(t) - C_\ell^{(j)}(s) &\leq \sum_{n'=n_1}^{n_2-1} x^{(j)}(\tau_{n'}) \\
&+ \left(x^{(j)}(\tau_{n_2}) \frac{t-\tau_{n_2}}{T_{n_2}} - x^{(j)}(\tau_{n_1}) \frac{s-\tau_{n_1}}{T_{n_1}} \right) \\
&+ 2(S_{\max}+1). \quad (49)
\end{aligned}$$

Thus, (14) follows from (47) and (49).

REFERENCES

[1] C. S. Chang, *Performance Guarantees in Communication Networks,* London: Springer-Verlag, 2000.

[2] D. B. West, *Introduction to Graph Theory*, 2nd ed., Prentice-Hall, Inc., 2001.

[3] C. -S. Chang, Y. -H. Hsu, J. Cheng, and D. -S. Lee, "A Dynamic Frame Sizing Algorithm for CICQ Switches with 100% Throughput, " *Proceedings of IEEE INFOCOM 2009.*

[4] P. Chaporkar and S. Sarkar, "Stable scheduling policies for maximizing throughput in generalized constrained queueing networks, " *Proceedings of IEEE INFOCOM 2006.*

[5] P. Giaccone, E. Leonardi, and D. Shah, "Throughput region of finite-buffered networks, " *IEEE Transaction on parallel and distributed systems,* vol. 18, no. 2, Feb. 2007.

[6] S.-M. He, S.-T. Sun, H.-T. Guan, Q. Zheng, Y.-J. Zhao, and W.Gao, "On guaranteed smooth switching for buffered crossbar switches," *IEEE/ACM Transactions on Networking*, vol. 16, no. 3, pp. 718-731, June 2008.

[7] L. B. Le, E. Modiano, and N. B. Shroff, "Optimal Control of Wireless Networks with Finite Buffers, " *Proceedings of IEEE INFOCOM 2010.*

[8] C. M. Lien and C. S. Chang, "Generalized Dynamic Frame Sizing Algorithm for Finite-Internal-Buffered Networks, " *IEEE Communication Letters,* vol. 13, no. 9, Sep. 2009.

[9] M. A. Marsan, A. Bianco, P. Giaccone, E. Leonardi, and F. Neri, "Multicast traffic in input-queued switches: optimal scheduling and maximum throughput, " *IEEE/ACM Transactions on Networking,* vol. 11, no. 3, June 2003.

[10] A. K. Parekh and R. G. Gallager, "A generalized processor sharing approach to flow control in integrated service networks: the multiple node case," *IEEE/ACM Transactions on Networking*, vol. 2, pp. 137-150, 1994.

[11] H. Sariowan, R.L. Cruz and G.C. Polyzos, "Scheduling for Quality of Service Guarantees via Service Curves," *Proceedings of the International Conference on Computer Communications and Networks*, 1995.

[12] G. Sharma, R. R. Mazumdar, and N. B. Shroff, "On the complexity of scheduling in wireless networks, " *MobiCom 2006,* Sep. 2006.

[13] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," *IEEE Transactions on Automatic Control*, vol. 31, no. 12, pp. 1936–1948, 1992