Flow-Level Stability of Multihop Wireless Networks Using Only MAC-Layer Information

Javad Ghaderi and R. Srikant
Department of ECE and Coordinated Science Lab.
University of Illinois at Urbana-Champaign
{jghaderi, rsrikant}@illinois.edu

***Abstract*—It is by now well-known that wireless networks with file arrivals and departures are stable if one uses α-fair congestion control and back-pressure based scheduling and routing. In this paper, we examine whether α-fair congestion control is necessary for flow-level stability. We show that stability can be ensured even with very simple congestion control mechanisms, such as a fixed window size scheme which limits the maximum number of packets that are allowed into the ingress queue of a flow. A key ingredient of our result is the use of the difference between the logarithms of queue lengths as the link weights. This result is reminiscent of results in the context of CSMA algorithms, but for entirely different reasons.**

I. Introduction

In order to operate wireless systems efficiently, scheduling algorithms are needed to facilitate simultaneous transmissions of different users. Scheduling algorithms for wireless networks have been widely studied since Tassiulas and Ephremides [1] proposed the *max weight* algorithm for single-hop wireless networks and its extension to multihop networks using the notion of *back-pressure* or *differential backlog*. The back-pressure algorithm (and hence, the max weight algorithm) is throughput optimal in the sense that it can stabilize the queues of the network for the largest set of arrival rates possible without knowing the actual arrival rates. The back-pressure algorithm works under very general conditions but it does not consider *flow-level dynamics*. It considers *packet-level dynamics* assuming that there is a fixed set of users/flows and packets are generated by each flow according to some stochastic process. In real networks, flows arrive randomly to the network, have only a finite amount of data, and depart the network after the data transfer is completed. Moreover, there is no notion of congestion control in the back-pressure algorithm while most modern communication networks use some congestion control mechanism for fairness purposes or to avoid excessive congestion inside the network [2].

There is a rich body of literature on the packet-level stability of scheduling algorithms, e.g., [1], [6], [7], [8]. Stability of wireless networks under flow-level dynamics has been studied in, e.g., [2], [3], [4]. Here, by stability, we mean that the number of flows in the network and the queue sizes at each node in the network remain finite. To achieve flow-level stability, these works use a specific way of congestion control based on *α-fair policies*; specifically, (a) the rates at which flows/files generate packets into their ingress queues maximize the sum-utility where each user has a utility function of the form $U(x) = x^{1-\alpha}/(1-\alpha)$ for some $\alpha > 0$ where x is the flow rate, and (b) the scheduling of packets in the network is performed based on the max wight/back-pressure algorithm.

When there are file/flow arrivals and departures, if the scheduler has access to the total queue length information at nodes, then it can use max weight/back-pressure algorithm to achieve throughput optimality, but this information is not typically available to the scheduler because it is implemented as part of the MAC layer. Moreover, without congestion control, queue sizes at different nodes could be widely different. This could lead to long periods of unfairness among flows.

Therefore, we need to use congestion control to provide better QoS. With congestion control, only a few packets from each file are released to the MAC layer at each time instant, and scheduling is done based on these MAC layer packets. However, prior work requires that a specific form of congestion control (namely, ingress queue-length based rate adaptation based on α-fair utility functions) has to be used. *Here we show that, in fact, very general window flow-control mechanisms are sufficient to ensure flow-level stability. The result suggests that ingress queue-based congestion control is more important than α-fairness to ensure network stability, when congestion control is used in conjunction with max weight scheduling/routing.*

In establishing the above result, we have used the max weight algorithm with link weights which are log-differentials of MAC-layer queue lengths. This observation is similar to the one made in the context of CSMA-type scheduling algorithms [9], [10], and it is interesting to note that we have arrived at the same choice of weight functions for entirely different reasons. Hence, this resolves the issue of distributed implementation of our scheduling algorithm using queue length-based CSMA algorithms. The results presented here are a significant extension of our earlier results in [12], where we only consider single hop flows.

The rest of the paper is organized as follows. In Section II, we describe our models for the wireless network, file arrivals, and Transport and MAC layers. We propose our scheduling

The research was supported in part by ARO MURIs W911NF-07-1-0287 and W911NF-08-1-0233 and AFOSR MURI FA 9550-10-1-0573.

algorithm in Section III. Section IV is devoted to the formal statement about the throughput-optimality of the algorithm and its proof sketch.

II. System Model

Model of wireless network

Consider a multihop wireless network consisting of a set of nodes $\mathcal{N} = \{1, 2, .., N\}$ and a set of links $\mathcal{L}$ between the nodes. There is a link from i to j, i.e., $(i, j) \in \mathcal{L}$, if transmission from i to j is allowed. Let $\mu = [\mu_{ij} : (i, j) \in \mathcal{L}]$ be the rates according to which links can transmit packets. Let $\mathcal{R}$ denote the set of available rate vectors (or transmission schedules) $\mathbf{r} = [r_{ij} : (i, j) \in \mathcal{L}]$. Note that each transmission schedule $\mathbf{r}$ corresponds to a set of node power assignments chosen by the network. Also let $\text{Co}(\mathcal{R})$ denote the convex hall of $\mathcal{R}$ which corresponds to time-sharing between different rate vectors. Hence, in general, $\mu \in \text{Co}(\mathcal{R})$. There are a set of users/sources $\mathcal{U} \subseteq \mathcal{N}$ and, for a user/source $u \in \mathcal{U}$, we use $d(u)(\neq u)$ to denote its destination. Let $\mathcal{D} := d(\mathcal{U})$ denote the set of destinations. We consider a time-slotted system. At each time slot t, new files can arrive at the source nodes and routing and scheduling decisions must be made to deliver the files to destinations in multihop fashion. We use $a_s(t)$ to denote the number of files that arrive at source s at time t and $\eta_s(f)$ to denote the size (number of packets) of file f, $1 \leq f \leq a_s(t)$. We assume that the process $\{a_s(t); s \in \mathcal{U}\}_{\{t \geq 0\}}$ is iid over time and independent across users with rate $[\lambda_s; s \in \mathcal{U}]$ and bounded second moments. Moreover, files that arrive at user s have the same size distribution independent of other users. Let $[\bar{\eta}_s; s \in \mathcal{U}]$ denote the vector of mean file sizes. We further assume that all file sizes are bounded from above by a constant B. Define $\rho_s := \lambda_s \bar{\eta}_s$ and let $\boldsymbol{\rho} = [\rho_s : s \in \mathcal{U}]$ be the vector of loads.

Model of Transport and MAC layers

Upon arrival of a file at a source Transport layer, a TCP-connection is established that regulates the injection of packets into the MAC layer. Once transmission of a file ends, file departs and the corresponding TCP-connection will be closed. MAC-layers are responsible for making the routing and scheduling decisions to deliver the MAC-layer packets to their destinations. Let $\mathcal{W}_{sf}(t)$ denote the congestion window size of file f at source s at time t. $\mathcal{W}_{sf}$ is a time-varying sequence which changes as a result of TCP congestion control. If the congestion window of file f is not full, TCP will continue injecting packets from the remainder of file f to the congestion window until file f has no packets remaining at the Transport layer or the congestion window becomes full. We consider ingress queue-based congestion control meaning that when a packet of congestion window departs the ingress queue, it is replaced with a new packet from its corresponding file at the Transport layer. It is important to note that the MAC layer does not know the number of remaining packets at the Transport layer, so scheduling and routing decisions have to be made based on the MAC-layers information only. In the rest of the paper, to expose the main idea, we consider a simple case where $\mathcal{W}_{sf}(t) \equiv 1$ as long as file f is present at source s. One can show that the results can actually be extended to general ingress queue-based congestion window dynamics when $1 \leq \mathcal{W}_{sf}(t) \leq \mathcal{W}_{max}$, i.e., all congestion window sizes are upper-bounded by a constant $\mathcal{W}_{max}$.

Queue dynamics

At the MAC layer of each node $n \in \mathcal{N}$, we consider per-destination queues $q_n^{(d)}$, $d \in \mathcal{D}$. A packet that is transmitted from one node to another node is removed from the MAC-layer queue of the first node and added to the MAC-layer queue of the second. Packet that reaches its destination is removed from the network. For the analysis, we use $Q_n^{(d)}$ (with capital Q) to denote the total per-destination queues. Note that if n is not a source, i.e., $n \notin \mathcal{U}$, then obviously $Q_n^{(d)} = q_n^{(d)}$ for all $d \in \mathcal{D}$. If n is source, we have to distinguish between the MAC-layer and total queues associated with the destination of node n because of packets existing at the Transport layer of n, i.e., $Q_n^{(d(n))} \neq q_n^{(d(n))}$. However, we still have $Q_n^{(d)} = q_n^{(d)}$ for all $d \in \mathcal{D} \backslash d(n)$. Let $x_{ij}^{(d)}$ denote the routing-plus-scheduling variable that shows the rate at which the packets of destination d are forwarded over the link (i, j). Then, the total-queue dynamics are given by

$$\begin{aligned} Q_n^{(d)}(t+1) &= \big(Q_n^{(d)}(t) - \sum_j x_{nj}^{(d)}(t)\big)_+ + \sum_i x_{in}^{(d)}(t) \\ &\quad + A_n^{(d)}(t), \end{aligned} \tag{1}$$

where $(\cdot)_+ := \max\{0, \cdot\}$ and $A_n^{(d)}(t) := \sum_{f=1}^{a_n(t)} \eta_n(f)$ is the total number of packets that new files bring to node n at time t and their destination is d. Note that we can write $\mathbb{E}\left[A_n^{(d)}(t)\right] = \rho_n^{(d)}$ where $\rho_n^{(d)} = \rho_n$ if $n \in \mathcal{U}$ and $d = d(n)$, and $\rho_n^{(d)} = 0$ otherwise. The capacity region of the network $\mathcal{C}$ is defined as the set of all load vectors $\boldsymbol{\rho}$ that under which the total-queues in the network can be stabilized. Note that under the connection-level model, stability of total-queues will imply that the number of files in the network is also stable. It is well-known [6] that a vector $\boldsymbol{\rho}$ belongs to $\mathcal{C}$ if and only if there exits a transmission rate vector $\mu \in \text{Co}(\mathcal{R})$ such that

$$\begin{aligned} &\mu_{ij}^{(d)} \geq 0; \ \forall d \in \mathcal{D} \text{ and } \forall (i, j) \in \mathcal{L}, \\ &\rho_n^{(d)} - \sum_j \mu_{nj}^{(d)} + \sum_i \mu_{in}^{(d)} \leq 0; \ \forall d \in \mathcal{D} \text{ and } \forall n \neq d, \\ &\sum_{d \in \mathcal{D}} \mu_{ij}^{(d)} \leq \mu_{ij}; \ \forall (i, j) \in \mathcal{L}. \end{aligned}$$

III. Description of Scheduling Algorithm

The algorithm is essentially the back-pressure algorithm [1] but it only uses the MAC-layer information. The key step in establishing the optimality of such an algorithm is using an appropriate weight function of the MAC-layer queues instead

of using the total queues. In particular, consider a weight function

$$g(x) := \log(1+x).$$

Let $C(n) = \{j : (n,j) \in \mathcal{L}\}$ denote the set of neighbors of node n. For each link $(i,j) \in \mathcal{L}$, define

$$w_{ij}^{(d)}(t) := g(q_i^{(d)}(t)) - g(q_j^{(d)}(t)).$$

Then the scheduling algorithm is as follows:

At each time t:

- Each node n observes the MAC-layer queue sizes of itself and its neighbors, i.e., $q_n^{(d)}$ and $q_j^{(d)}$ for $j \in C(n)$ and for all $d \in \mathcal{D}$.
- For each link (i,j), find

$$d_{ij}^*(t) = \arg\max_{d\in\mathcal{D}} w_{ij}^d(t).$$

- For each link (i,j), define a weight

$$w_{ij}(t) := \max_{d\in\mathcal{D}} w_{ij}^d(t).$$

  The network needs to find the optimal transmission schedule $\tilde{x}^* \in \mathcal{R}$ that solves

$$\max_{r\in\mathcal{R}} \sum_{(i,j)\in\mathcal{L}} r_{ij} w_{ij}(t).$$

- Finally, assign $x_{ij}^{(d)}(t) = \tilde{x}_{ij}^*$ if $d = d_{ij}^*(t)$, and zero otherwise (break ties at random).

IV. System Stability

Order of events

Since we use a discrete-time model, we have to specify the order in which files/packets arrive and depart, which we do below:

1) At the beginning of each time slot, a scheduling decision is made by the scheduling algorithm. Packets depart from the MAC layer of scheduled links.
2) File arrivals occur next. Once a file arrives, a new TCP connection is set up for that file with an initial pre-determined congestion window size.
3) For each TCP connection, if the congestion window is not full, packets are injected into the MAC layer from the Transport layer until the window size is fully used or there is no more packets at the Transport layer.

State of the system

Define the state of node n as

$$\begin{aligned}\mathcal{S}_n(t) \quad = \quad & \big\{(q_n^{(d)}(t), \mathcal{I}_n^{(d)}(t)) : d \in \mathcal{D}, \\ & (U_{nf}(t), \mathcal{W}_{nf}(t)) : 1 \le f \le N_n(t)\big\},\end{aligned}$$

where $N_n(t)$ is the number of existing files at node n at time t, $U_{nf}(t)$ is the number of remaining packets of file f at node n at time t, and $\mathcal{W}_{nf}(t)$ is its corresponding congestion window size. Obviously, if n is not a source node, then we can remove $(U_{nf}, \mathcal{W}_{nf})$ from the description of $\mathcal{S}_n$. $\mathcal{I}_n^{(d)}(t)$ denotes the information required about $q_n^{(d)}(t)$ to serve the MAC-layer packets which depends on the specific service discipline implemented in MAC-layer queues. In the rest of the paper, we consider the case of FIFO service discipline in MAC-layer queues and $\mathcal{W}_{nf}(t) = 1$. In this case, $\mathcal{I}_n^{(d)}(t)$ is simply the ordering of packets in $q_n^{(d)}(t)$ according to their entrance times. As it will turn out from the proof, the system stability will hold for any none-idling service discipline. Define the state of the system to be $\mathcal{S}(t) = \{\mathcal{S}_n(t) : n \in \mathcal{N}\}$. Now, given the scheduling algorithm in section III, $\mathcal{S}(t)$ evolves as a discrete-time Markov chain. Next, we show that this Markov chain is positive recurrent.

Theorem 1. *For any $\boldsymbol{\rho}$ strictly inside $\mathcal{C}$, the scheduling algorithm, in Section III, can stabilize the network independent of Transport-layer ingress queue-based congestion control mechanism (as long as the minimum window size is one and the window sizes are bounded) and the (nonidling) service discipline used to transmit packets from active nodes.*

The rest of the section is devoted to the proof of the above theorem.

Proof of Theorem 1

Let $G(u) := \int_0^u g(x)dx$, where $g(x) = \log(1+x)$. Then G is a strictly convex function. Consider a Lyapunov function

$$V(\mathcal{S}(t)) = \sum_{n=1}^{N} \sum_{d\in\mathcal{D}} G(Q_n^{(d)}(t)).$$

Note that given the state $\mathcal{S}(t)$, $Q_n^{(d)}$ is known. Let $\Delta V(t) := V(\mathcal{S}(t+1)) - V(\mathcal{S}(t))$, i.e.,

$$\Delta V(t) = \sum_{n=1}^{N} \sum_{d\in\mathcal{D}} \big\{G(Q_n^{(d)}(t+1)) - G(Q_n^{(d)}(t))\big\},$$

then, using convexity of G, we get

$$\Delta V(t) \le \sum_{n=1}^{N} \sum_{d\in\mathcal{D}} g(Q_n^{(d)}(t+1))(Q_n^{(d)}(t+1) - Q_n^{(d)}(t)).$$

Let r_{max} denote the maximum link capacity over all the links, then observe that

$$\begin{aligned}|Q_n^{(d)}(t+1) - Q_n^{(d)}| \quad \le \quad & \max\Bigg\{A_n^{(d)}(t) + \sum_{i:(i,n)\in\mathcal{L}} x_{in}^{(d)}(t), \\ & \qquad , \sum_{j:(n,j)\in\mathcal{L}} x_{nj}^{(d)}(t)\Bigg\},\end{aligned}$$

and hence

$$|Q_n^{(d)}(t+1) - Q_n^{(d)}| \le A_n^{(d)}(t) + N r_{max}.$$

Using the concavity of g and the fact that $g' \le 1$, it can be shown that

$$|g(\bar{Q}_n^{(d)}(t+1)) - g(\bar{Q}_n^{(d)}(t))| \le |\bar{Q}_n^{(d)}(t+1) - \bar{Q}_n^{(d)}(t)|.$$

Hence,

$$\begin{aligned}\Delta V(t) &\leq \sum_{n=1}^{N}\sum_{d\in\mathcal{D}} g\big(Q_n^{(d)}(t)\big)\big(Q_n^{(d)}(t+1)-Q_n^{(d)}(t)\big) \\ &\quad+\sum_{n=1}^{N}\sum_{d\in\mathcal{D}}\Big\{|g(\bar{Q}_n^{(d)}(t+1))-g(\bar{Q}_n^{(d)}(t))|\times \\ &\qquad\qquad |Q_n^{(d)}(t+1)-Q_n^{(d)}(t)|\Big\} \\ &\leq \sum_{n=1}^{N}\sum_{d\in\mathcal{D}} g\big(Q_n^{(d)}(t)\big)\big(Q_n^{(d)}(t+1)-Q_n^{(d)}(t)\big) \\ &\quad+\sum_{n=1}^{N}\sum_{d\in\mathcal{D}}(A_n^{(d)}(t)+Nr_{max})^2.\end{aligned}$$

Therefore,

$$\begin{aligned}\Delta V(t) &\leq \sum_{n=1}^{N}\sum_{d\in\mathcal{D}}\Bigg\{g(Q_n^{(d)}(t))[\sum_{i:(i,n)\in\mathcal{L}} x_{in}^{(d)}(t)+A_n^{(d)}(t) \\ &\quad-\sum_{j:(n,j)\in\mathcal{L}} x_{nj}^{(d)}(t)]\Bigg\} \\ &\quad+\sum_{n=1}^{N}\sum_{d\in\mathcal{D}} g(Q_n^{(d)}(t))\xi(Q_n^{(d)}(t),\{x_{ij}^{(d)}(t)\}) \\ &\quad+\sum_{n=1}^{N}\sum_{d\in\mathcal{D}}(A_n^{(d)}(t)+Nr_{max})^2,\end{aligned}$$

where,

$$\begin{aligned}\xi(Q_n^{(d)},\{x_{ij}^{(d)}\}) &:= \big(Q_n^{(d)}-\sum_{j:(n,j)\in\mathcal{L}} x_{nj}^{(d)}\big)_+ \\ &\quad-(Q_n^{(d)}-\sum_{j:(n,j)\in\mathcal{L}} x_{nj}^{(d)}).\end{aligned}$$

Note that if $Q_n^{(d)}(t) > \sum_{j:(n,j)\in\mathcal{L}} x_{nj}^{(d)}(t)$, then $\xi(Q_n^{(d)},\{x_{ij}^{(d)}\}) = 0$. If $Q_n^{(d)} \leq \sum_{j:(n,j)\in\mathcal{L}} x_{nj}^{(d)}$, then $0 \leq \xi(Q_n^{(d)},\{x_{ij}^{(d)}\}) \leq \sum_{j:(n,j)\in\mathcal{L}} x_{nj}^{(d)}(t) \leq Nr_{max}$.

Hence,

$$\begin{aligned}\Delta V(t) &\leq \sum_{n=1}^{N}\sum_{d\in\mathcal{D}} g(Q_n^{(d)}(t))[\sum_{i:(i,n)\in\mathcal{L}} x_{in}^{(d)}(t)+A_n^{(d)}(t) \\ &\quad-\sum_{j:(n,j)\in\mathcal{L}} x_{nj}^{(d)}(t)]+N^3 r_{max}g(Nr_{max}) \\ &\quad+\sum_{n=1}^{N}\sum_{d\in\mathcal{D}}(A_n^{(d)}(t)+Nr_{max})^2.\end{aligned}$$

Define $\mathbb{E}_{\mathcal{S}(t)}[.] = \mathbb{E}[.|\mathcal{S}(t)]$. Taking the expectation of both sides, given the state at time t is known, yields

$$\begin{aligned}\mathbb{E}_{\mathcal{S}(t)}[\Delta V(t)] &\leq C_1+\sum_{n=1}^{N}\sum_{d\in\mathcal{D}}\{g(Q_n^{(d)}(t))\mathbb{E}_{\mathcal{S}(t)}[\rho_n^{(d)} \\ &\quad+\sum_{i:(i,n)\in\mathcal{L}} x_{in}^{(d)}(t)-\sum_{j:(n,j)\in\mathcal{L}} x_{nj}^{(d)}(t)]\},\end{aligned}$$

for a constant

$$\begin{aligned}C_1 &= N^3 r_{max}f(Nr_{max})+\mathbb{E}\left[\sum_{n=1}^{N}\sum_{d\in\mathcal{D}}(A_n^{(d)}(t)+Nr_{max})^2\right] \\ &< \infty\end{aligned}$$

because $\mathbb{E}\left[A_n^{(d)}(t)^2\right] < \infty$. Changing the order of summations, we have

$$\begin{aligned}\mathbb{E}_{\mathcal{S}(t)}[\Delta V(t)] &\leq C_1+\sum_{n=1}^{N}\sum_{d\in\mathcal{D}} g(Q_n^{(d)}(t))\rho_n^{(d)} \\ &\quad-\mathbb{E}_{\mathcal{S}(t)}\left[\sum_{(i,j)\in\mathcal{L}}\sum_{d\in\mathcal{D}} x_{ij}^{(d)}(t)\big(g(Q_i^{(d)}(t))-g(Q_j^{(d)}(t))\big)\right].\end{aligned}$$

Recall that, at each node n, for all destinations $d \neq d(n)$, we have $Q_n^d = q_n^d$ where q_n^d denotes the MAC-layer queue at node n destined for d. Also recall that file sizes are bounded, i.e., if n is a source, $U_{nf}(t) \leq B < \infty$ for any existing file f at n. If $d = d(n)$ is the destination of n, then Q_n^d consists of: (i) packets received from upstream flows destined for d, and (ii) MAC-layer packets received from the files generated at n itself. In the simple case that $\mathcal{W}_{nf}(t) = 1$, q_n^d shows the number of files in the network that are intended for destination d and are generated at node n or use n as an intermediate relay. Therefore, it is clear that $q_n^d \leq Q_n^d \leq q_n^d \times B$. Hence for all n and d,

$$g(q_n^d) \leq g(Q_n^d) \leq g(q_n^d)+g(B).$$

Define two types of link weights based on the total queues and MAC-layer queues respectively as

$$W_{ij}^{(d)} := g(Q_i^d)-g(Q_j^d),$$

and,

$$w_{ij}^{(d)} := g(q_i^d)-g(q_j^d).$$

Then, it follows that

$$|W_{ij}^{(d)}-w_{ij}^{(d)}| \leq g(B);\ \forall(i,j)\in\mathcal{L}, d\in\mathcal{D}. \tag{2}$$

Let

$$W_{ij} := \max_d W_{ij}^{(d)},$$

and,

$$w_{ij} := \max_d w_{ij}^{(d)}.$$

Also let

$$d_1^* = \arg\max_d W_{ij}^{(d)},$$

and,

$$d_2^* = \arg\max_d w_{ij}^{(d)}.$$

Then, $\forall (i,j) \in \mathcal{L}$,

$$w_{ij} \geq w_{ij}^{(d_1^*)} \geq W_{ij} - g(B),$$

and,

$$W_{ij} \geq W_{ij}^{(d_2^*)} \geq w_{ij} - g(B).$$

This shows that $|W_{ij} - w_{ij}| \leq g(B)$.

Let x^* be the max weight schedule based on weights W_{ij}, i.e.,

$$x^* = \arg\max_{x \in \mathcal{R}} \sum_{(i,j)\in\mathcal{L}} x_{ij} W_{ij}.$$

Similarly let $\tilde{x}^*$ denote the max weight schedule based on MAC-layer queues, i.e.,

$$\tilde{x}^* = \arg\max_{x \in \mathcal{R}} \sum_{(i,j)\in\mathcal{L}} x_{ij} w_{ij}.$$

Then, the weight of $\tilde{x}^*$ differs from the weight of x^* only by a constant for all queue values as we show next. First note that from the definition of x_{ij}^*,

$$\sum_{(i,j)\in\mathcal{L}} x_{ij}^* W_{ij} - \sum_{(i,j)\in\mathcal{L}} \tilde{x}_{ij}^* W_{ij} \geq 0.$$

On the other hand,

$$\begin{aligned}
&\sum_{(i,j)\in\mathcal{L}} x_{ij}^* W_{ij} - \sum_{(i,j)\in\mathcal{L}} \tilde{x}_{ij}^* W_{ij} \\
&= \sum_{(i,j)\in\mathcal{L}} x_{ij}^* W_{ij} - \sum_{(i,j)\in\mathcal{L}} x_{ij}^* w_{ij} \qquad (3)\\
&+ \sum_{(i,j)\in\mathcal{L}} x_{ij}^* w_{ij} - \sum_{(i,j)\in\mathcal{L}} \tilde{x}_{ij}^* w_{ij} \qquad (4)\\
&+ \sum_{(i,j)\in\mathcal{L}} \tilde{x}_{ij}^* w_{ij} - \sum_{(i,j)\in\mathcal{L}} \tilde{x}_{ij}^* W_{ij} \qquad (5)\\
&\leq 2N^2 r_{max} g(B),
\end{aligned}$$

because (3) and (5) are less than $N^2 r_{max} g(B)$ each and (4) is negative by definition of $\tilde{x}^*$. Hence, under MAC scheduling $\tilde{x}^*$, the Lyapunov drift is bounded as follows.

$$\begin{aligned}
&\mathbb{E}_{\mathcal{S}(t)}\left[\Delta V(t)\right] \leq C_2 + \sum_{n=1}^{N}\sum_{d\in\mathcal{D}} g(Q_n^{(d)}(t))\rho_n^{(d)} - \\
&\mathbb{E}_{\mathcal{S}(t)}\left[\sum_{(i,j)\in\mathcal{L}} x_{ij}^*(t)(g(Q_i^{(d^*)}(t)) - g(Q_j^{(d^*)}(t)))\right] \\
&= C_2 + \sum_{n=1}^{N}\sum_{d\in\mathcal{D}}\Bigg\{ g(Q_n^{(d)}(t))\mathbb{E}_{\mathcal{S}(t)}[\rho_n^{(d)} + \sum_{i:(i,n)\in\mathcal{L}} {x^*}_{in}^{(d)}(t) \\
&- \sum_{j:(n,j)\in\mathcal{L}} {x^*}_{nj}^{(d)}(t)]\Bigg\},
\end{aligned}$$

where $C_2 = C_1 + 2N^2 r_{max} g(B)$. The rest of the proof is standard. Since load $\boldsymbol{\rho}$ is strictly inside the capacity region, there must exist a $\epsilon > 0$ and a $\mu \in \mathrm{Co}(\mathcal{R})$ such that

$$\rho_n^{(d)} + \epsilon \leq \sum_{j:(n,j)\in\mathcal{L}} \mu_{nj}^{(d)} - \sum_{i:(i,n)\in\mathcal{L}} \mu_{in}^{(d)} \; ; \forall n \in \mathcal{N}, \forall d \in \mathcal{D}. \quad (6)$$

Hence, for any $\delta > 0$,

$$\begin{aligned}
&\mathbb{E}_{\mathcal{S}(t)}\left[\Delta V(t)\right] \leq \\
&\sum_{n=1}^{N}\sum_{d\in\mathcal{D}} g(Q_n^{(d)}(t))\left[\sum_{i:(i,n)\in\mathcal{L}} {x^*}_{in}^{(d)}(t) - \sum_{j:(n,j)\in\mathcal{L}} {x^*}_{nj}^{(d)}(t)\right] \\
&- \sum_{n=1}^{N}\sum_{d\in\mathcal{D}} g(Q_n^{(d)}(t))\left[\sum_{i:(i,n)\in\mathcal{L}} \mu_{in}^{(d)}(t) - \sum_{j:(n,j)\in\mathcal{L}} \mu_{nj}^{(d)}(t)\right] \\
&- \epsilon\sum_{n=1}^{N}\sum_{d\in\mathcal{D}} g(Q_n^{(d)}(t)) + C_2 \\
&\leq -\epsilon\sum_{n=1}^{N}\sum_{d\in\mathcal{D}} g(Q_n^{(d)}(t)) + C_2 \leq -\delta,
\end{aligned}$$

whenever $\max_{n,d} Q_n^{(d)} \geq g^{-1}\big(\frac{C_2+\delta}{\epsilon}\big)$ or, as a sufficient condition, whenever $\max_{n,d} q_n^{(d)} \geq g^{-1}\big(\frac{C_2+\delta}{\epsilon}\big)$. Specifically, consider a set

$$\mathcal{B} = \big\{\mathcal{S} : \max_{n,d} q_n^{(d)} \geq g^{-1}\big(\frac{C_2+\delta}{\epsilon}\big)\big\}.$$

Then, for any $\mathcal{S} \in \mathcal{B}$, the Lyapunov drift is less than $-\delta$. Also it is easy to check that $\mathcal{B}^c$ contains only a finite set of states with finite drift.Therefore, it follows that the system is stable by Foster-Lyapunov stability theorem [11]. In particular, queue sizes and the number of files in the system are stable.

Remark 1. *The proof argument essentially works for very general ingress queue-based congestion control mechanisms where the congestion window sizes are at least one and bounded from above by a constant. We only require that the congestion window dynamics could be described as a function of queue lengths of the network so that the network Markov chain is well-defined. Even in the case that the congestion window is a function of the delayed queue lengths of the network up to T time slots before, e.g., due to the feedback delay of at most T from destination to source, the network state could be modified, to include the queues up to T time slots before, so that the same proof technique still applies.*

V. Conclusions

Design of efficient scheduling and congestion control algorithms can be decoupled by using MAC-layer queues for the scheduling of packets and using window-based congestion control mechanisms for controlling the rate at which packets injected into the network. A key ingredient of our result is the use of the difference between the logarithms of queue lengths

as the link weights. This separation result is appealing to the network operator. It is also important from the practical perspective because, typically, only the MAC-layer information is available to the scheduler since it is implemented as part of the MAC layer.

REFERENCES

[1] L. Tassiulas and A. Ephremides, Stability properties of constrained queueing systems and scheduling algorithms for maximal throughput in multihop radio networks, *IEEE Transactions on Automatic Control*, vol. 37, no. 12, pp. 1936-1948, 1992.

[2] X. Lin, N. Shroff, and R. Srikant, On the connection-level stability of congestion-controlled communication networks, *IEEE Transactions on Information Theory*, vol. 54, no. 5, pp. 2317-2338, 2008.

[3] J. Liu, A. Proutiere, Y. Yi, M. Chiang, and V. Poor, flow-level stability of data networks with non-convex and time-varying rate regions, *Proc. ACM SIGMETRICS*, 2007.

[4] C. Moallemi and D. Shah, On the flow-level dynamics of a packet-switched network, *Proc. ACM SIGMETRICS*, pp. 83-94, June 2010.

[5] X. Lin, N. Shroff and R. Srikant, A tutorial on cross-layer optimization in wireless networks, *IEEE Journal on Selected Areas in Communications*, August 2006, pp. 1452-1463.

[6] M. J. Neely, E. Modiano, and C. E. Rohrs, Dynamic power allocation and routing for time varying wireless networks, *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 1, pp. 89-103, January 2005.

[7] I. Keslassy and N. McKeown, Analysis of scheduling algorithms that provide 100% throughput in input-queued switches, *Proc. Allerton Conference on Communication, Control, and Computing*, 2001.

[8] A. Eryilmaz, R. Srikant, and J. R. Perkins, Stable scheduling algorithms for fading wireless channels. *IEEE/ACM Transactions on Networking*, vol. 13, no. 2, pp. 411-424, April 2005.

[9] J. Ghaderi and R. Srikant, On the design of efficient CSMA algorithms for wireless networks, *IEEE Conference on Decision and Control*, 2010.

[10] S. Rajagopalan, D. Shah and J. Shin, Network adiabatic theorem: an efficient randomized protocol for contention resolution, *ACM SIGMETRICS/Performance*, pp. 133-144, 2009.

[11] S. Asmussen, Applied probability and queues, Springer, 2003.

[12] J. Ghaderi, T. Ji, and R. Srikant, Connection-level scheduling in wireless networks using only MAC-layer information, *IEEE INFOCOM 2012 Mini-Conference*, accepted.