End-to-end bandwidth guarantees through fair local spectrum share in wireless ad-hoc networks

Saswati Sarkar and Leandros Tassiulas

Abstract—**Sharing the locally common spectrum among the links of the same vicinity is a fundamental problem in wireless ad-hoc networks. Lately some scheduling approaches have been proposed that guarantee fair share of the bandwidth among the links. What really affects the quality of service perceived by the applications though, is the effective end-to-end bandwidth allocated to the different network sessions that span several links of such a wireless ad-hoc network. In the current paper we propose an algorithm that provides fair session rates in that context. The algorithm is based on a combination of a link scheduling method to avoid local conflicts, a fair session service discipline per link and a hop-by-hop window flow control scheme. It is proven in the paper that the long term rates allocated to the different sessions are maxmin fair. All the stages of the algorithm are implementable based on local information only, except of the link scheduling part that needs some network-wide coordination of the links. When the latter is replaced by an approximate distributed link scheduling algorithm then we may have a fully distributed solution with suboptimal performance. Some numerical study is performed to evaluate the impact of various parameter choices on the performance of the algorithm.**

I. Introduction

Link transmission scheduling in multihop wireless networks attracted a lot of attention over the last twenty years since it is an issue of fundamental importance for the proper operation of those systems. The earlier work on the subject was focused mostly on obtaining network operating algorithms that were guaranteeing end-to-end connectivity whenever that was feasible such that end-to-end information transport could be supported [1], [2], [9], [13]. As the applications became more and more bandwidth hungry as well as sensitive to the perceived quality of service there has been a lot of effort lately to obtain transmission scheduling algorithms that provide some guarantees on the effective rates enjoyed by each individual link [4], [7], [8], [16], [14], [19]. A quantitative framework that has been adopted widely for formulating the rate guarantees is that of maxmin fair rate allocation. In [19] an algorithm is proposed that provably provides maxmin fair rates to the wireless links. Similarly, most of the previous work on the subject is focused on guaranteeing certain rates at the link or single-hop session level as well. What really affects the quality of service perceived by the applications though, is the effective end-to-end bandwidth allocated to the different network sessions that span several links of such a wireless ad-hoc network. In the current paper we propose an algorithm that provides fair session rates in that context. We would like to mention that [10] addresses fairness in context of multihop sessions as well. However, the fairness objective there is to distribute the bandwidth among sessions in accordance with pre-specified weights, while our objective is to attain a maxmin fair allocation of bandwidth via scheduling without any pre-computation of the fair shares. Thus a scheduling framework like [10] which needs knowledge of desired ratios between the bandwidth allocation of different sessions does not apply to our case.

Providing end-to-end rate guarantees in wired networks is a subject of great importance, that has been studied extensively [5][12]. In the current work we borrow some features of network control approaches that have been proposed for wired networks and we combine them with wireless link scheduling techniques to come-up with an algorithm that provides end-to-end guarantees in a wireless setting. The algorithm is based on a combination of a link scheduling method to avoid local conflicts, a fair session service discipline per link and a hop-by-hop window flow control scheme. It is proven in the paper that the long term rates allocated to the different sessions are maxmin fair. Regarding the implementation of the algorithm, most of its stages are implementable based on information that is available locally in the node where the scheduling is performed. The only exception is the link scheduling part that needs some network-wide coordination of the links. The link scheduling needs to compute a maximum weighted matching of the networks topology graph. When the latter computation is replaced by an approximate distributed link scheduling algorithm then we may have a fully distributed solution with suboptimal performance. The proposed algorithm is evaluated numerically and the impact of various parameter choices on the performance of the algorithm is discussed.

The paper is organized as follows. We explain the fair-

S. Sarkar is with the department of Electrical and Systems Engineering in the University of Pennsylvania. Email: swati@ee.upenn.edu

L. Tassiulas is with the department of Electrical and Computer Engineering in the University of Maryland, College Park. Email: leandros@isr.umd.edu

ness objectives and the network model in Section II. We present scheduling strategies which attain maxmin fairness and a more generalized notion of weighted maxmin fairness in Section III. Section III also presents the analytical performance guarantees of these algorithms. We investigate the impact of the choice of certain algorithm parameters via simulation in Section IV. We discuss several implementation related features of our algorithms in Section V. We present the proof of the analytical results in section VI. The proof offers significant insight about the overall functioning of the algorithms.

II. Fairness Objectives and Network Model

Symbol	Meaning
N	Number of sessions
r_i	Maxmin fair rate of session i
ρ_i	Packet generation rate of session i
G_i	Weight of session i
W	Window Parameter
L_i	Path length of session i
L	Maximum path length of the sessions
F	Number of distinct ranks in the system
W	Window Parameter
$S_{i,n}(t)$	Number of times session i has been sampled at node n by time t
$C_{i,n}(t)$	Number of tokens of session i at node n at time t
$C_{i,n}^{\text{unused}}(t)$	Number of unused tokens of session i at node n at time t
$P_{i,n}(t)$	Number of released packets of session i at node n at time t
$D_i(t)$	Number of packets of session i delivered to the destination at time t

TABLE I
SUMMARY OF SYMBOLS WHICH HAVE BEEN USED REPEATEDLY THROUGHOUT THE PAPER

We consider a network with V nodes and E wireless links. There are N sessions. Each session has a source and a destination node. The route from the source to the destination may include several other nodes. We assume that the routes are predetermined, and do not consider the routing problem in this paper. We assume that every node has one radio unit. The implication is that a node can be involved in a single transmission at a time, i.e., a node can either transmit one packet or receive one packet or remain idle. We assume that every node has a locally unique frequency. This means that multiple transmissions can proceed simultaneously without any interference as long as they do not have a common node. Thus the links which are active at any time must constitute a matching. For example, a bluetooth network satisfies the above assumptions[11]. However, we do not consider bluetooth specifically, but rather our framework applies to any wireless network which satisfies the above assumptions.

The focus of this paper is on investigating fairness issues in multihop sessions in presence of the wireless scheduling constraints described above. We do not consider channel errors in this paper. Now we describe how the scheduling constraints determine the feasible set of bandwidth allocations. A bandwidth allocation $(r_1, \ldots, r_N)$ can be attained if there exists a scheduling sequence which attains the corresponding rates. Assume that all nodes can transmit packets at the same rate, i.e., normalizing the bandwidth, every node can transmit one packet per unit time. Hajek *et al.*[6] showed that if the network is a bipartite graph* then a bandwidth allocation $(r_1, \ldots, r_N)$ is attainable if and only if the sum of bandwidth of all sessions traversing a node is less than or equal to 1. Many wireless networks are indeed bipartite graphs. Once again, bluetooth networks are often bipartite. For non-bipartite graphs there is a simple condition for guaranteeing stability[6]. The result is that a bandwidth allocation $(r_1, \ldots, r_N)$ can be attained if the sum of bandwidth of all sessions traversing a node is less than or equal to $2/3$†. This result is sufficient for our purpose, as in practice, bandwidth is allocated so as to utilize the bandwidth capacity of the nodes only partially, and leave the rest for protection against transients and overloads. Thus, combining the two cases, we will consider a bandwidth allocation $(r_1, \ldots, r_N)$ to be attainable if the sum of bandwidth of all sessions traversing a node is less than or equal to α, where α is the desired bandwidth utilization factor ($\alpha \leq 2/3$ for non-bipartite graphs and $\alpha \leq 1$ for bipartite graphs). We refer to this constraint as the *node capacity constraint.*

If a session i generates packets at the rate ρ_i, then its bandwidth r_i is upper bounded by ρ_i, $r_i \leq \rho_i$. We refer to this constraint as the *demand constraint.*

A bandwidth allocation is feasible if and only if it satisfies both the demand and the node capacity constraints.

We define the notion of maxmin fairness. A feasible bandwidth allocation is maxmin fair if the bandwidth of a session can not be increased and feasibility be retained without decreasing that of another session which has equal or lower bandwidth. More formally, a feasible bandwidth allocation $(r_1, r_2, \ldots, r_N)$ is maxmin fair, if it satisfies the

*A bipartite graph is one where the vertex set can be partitioned in two sets such that there is no edge connecting the vertices in the same set.

†The result is sufficient, but not necessary, i.e., a bandwidth allocation can be attained even if this condition is not satisfied.

following property w.r.t. any other feasible bandwidth allocation $(s_1, \ldots, s_N)$: if there exists a i such that $r_i < s_i$, then there exists a j such that $r_j \leq r_i$ and $s_j < r_j$.

We now present a necessary and sufficient condition for maxmin fairness. We need to introduce the concept of bottleneck nodes for this purpose. *A node is a bottleneck node of a session if the bandwidth allocated to the session is the maximum among the bandwidth allocated to all sessions traversing the node and the sum of the bandwidth of sessions traversing a node is equal to the bandwidth utilization factor α.*

We state without proof the following lemma which presents a necessary and sufficient condition for maxmin fairness of multihop sessions.

Lemma 1: The following condition is necessary and sufficient for a feasible bandwidth allocation to be maxmin fair: for every session i, either the bandwidth allocated to session i is equal to its long term packet generation rate or the session has a bottleneck node.

The intuition behind the result is as follows. The bandwidth allocated to a session can not be increased beyond its packet generation rate. If however the bandwidth allocated to a session i is less than its packet generation rate, then its bandwidth can not be increased without reducing that of any other session traversing its bottleneck node, on account of the node capacity constraint at its bottleneck node. This means that the bandwidth of a session which has bandwidth less than or equal to that of i, must be decreased in order to accommodate an increase in the bandwidth of session i. This shows that the bandwidth allocation is maxmin fair. The bottleneck condition is similar to one that holds for wireline sessions[3].

The basic principle behind maxmin fair allocation is to discriminate between sessions only on the basis of congestion and their packet generation rates. If two sessions traverse paths of similar congestion and have the same traffic demands, then both would be allocated the same bandwidth under maxmin fair allocation. A more generalized notion would be to provide a "weighted" maxmin fairness, where bandwidth is allocated in accordance to given weights for sessions. We describe the concept of weighted maxmin fairness next.

Every session i has weight G_i. Consider a bandwidth allocation $(r_1, \ldots, r_N)$. Weighted bandwidth of a session i is r_i/G_i. A bandwidth allocation is maxmin fair if the weighted bandwidth of a session can not be increased without decreasing that of another session which has equal or lower value of the weighted bandwidth. More formally, a feasible bandwidth allocation $(r_1, r_2, \ldots, r_N)$ is maxmin fair, if it satisfies the following property w.r.t. any other feasible bandwidth allocation $(s_1, \ldots, s_N)$: if there exists a i such that $r_i < s_i$, then there exists a j such that $r_j/G_j \leq r_i/G_i$ and $s_j < r_j$.

We now present a similar bottleneck condition for weighted maxmin fairness. *A node is a weighted bottleneck node of a session if the weighted bandwidth allocated to the session is the maximum among the weighted bandwidth allocated to all sessions traversing the node and the sum of the bandwidth of the sessions traversing a node is equal to the bandwidth utilization factor α.* The following lemma presents a necessary and sufficient condition for weighted maxmin fairness of multihop sessions.

Lemma 2: A feasible bandwidth allocation is weighted maxmin fair if and only if, for every session i, either the bandwidth allocated to session i is equal to its long term packet generation rate or the session has a weighted bottleneck node.

The intuition behind this necessary and sufficient condition is similar to the previous one.

III. A BACK-PRESSURE BASED FAIR BANDWIDTH ALLOCATION ALGORITHM

We design an algorithm to attain maxmin fairness for multihop sessions. We consider a two-tier approach for this problem. The first step computes the maxmin fair bandwidth share of a session in each node on its path, and releases packets for transmission in accordance with these fair shares. The second step schedules the transmission of the released packets so as to attain the fair shares. This modularization enables us to use different algorithms for attaining different fairness objectives (e.g., maxmin fairness, weighted maxmin fairness etc.) for the first part and attain the bandwidth shares computed as per the desired objective using an existing scheduling strategy[18] in the second part. This scheduling, denoted as the maximum difference backlog based scheduling can stabilize the network for any feasible arrival process, and hence can attain any feasible bandwidth allocation in wireless networks. Since the packet release process is fair and hence feasible, the overall framework attains maxmin fairness. The key contribution of this paper is the first step. We present the basic algorithm and its performance guarantees in subsection III-A. We consider generalizations in subsections III-B and III-C. We have defined several recurring symbols in symbol table I.

A. Basic Algorithm

We consider a saturated case, i.e., the source node of each session has an infinite supply of packets ($\rho_i = \infty$ for each session i). In this case feasibility and maxmin fairness are determined by the node capacity constraints alone, and the demand constraints do not exist (page 2). The algorithm has been presented in Figure 1. Here we describe

Procedure Token Generation(node m)
begin

Sample each session traversing node m in round robin order.
Let session i traverse node m and
nodes l, n be adjacent to node m in the path of session i,
When session i is sampled in slot t:
if $C_{i,m}(t) < C_{i,l}(t) + W$ and $C_{i,m}(t) < C_{i,n}(t) + W$ **then**
Generate a token for session i in slot t ($C_{i,m}(t+1) = C_{i,m}(t) + 1$;
else
Do not generate token for session i ($C_{i,m}(t+1) = C_{i,m}(t)$) and
Sample the next session in the round robin order

end

Procedure Packet Release(source i)
begin

Release a new session i packet for transmission at session i source when a new token is generated at the source.

end

Procedure Packet Scheduling For Transmission(link e)
begin

Let link e be between nodes m and n,
let $\mathcal{L}_e$ be the set of sessions traversing link e and
$P_{i,n}(t)$ is the number of released packets of session i waiting at node n.
$W_e(t) = \max_{i \in \mathcal{L}_e}(P_{i,m}(t) - P_{i,n}(t))$ (/*Weight of link is the maximum difference in backlog across the link */)
Schedule the links which constitute a maximum weighted matching
If link e is scheduled then,
transmit a packet of session j from node m to node n if
$P_{j,m}(t) - P_{j,n}(t) = \max_{i \in \mathcal{L}_e}(P_{i,m}(t) - P_{i,n}(t))$ /*session j has the maximum difference of backlog across e*/

end

Fig. 1. Pseudo code of the fair bandwidth allocation algorithm for saturated sessions

each part. Fair bandwidth is computed by a token generation process. Every node generates tokens for all sessions traversing the node. The token generation process is so designed that the tokens are generated for each session at its maxmin fair rate. Whenever a new token is generated for a session at its source node, the source node releases a new packet for transmission. Thus the packet release process is maxmin fair as well. Only the "released" packets are eligible for transmission, i.e., whenever a source has an opportunity to transmit, it transmits only the released packets.

We now describe the token generation process, and thereby present the intuition why the token generation equals the maxmin fair rate. The maxmin fair rate of a session is determined by the bandwidth offered by its bottleneck node, which happens to be the most congested node on its path. Intuitively, the token generation rate for a session at any node on its path should equal that at the bottleneck node. The challenge is to attain this at every node on the path of a session without explicit information about the bottleneck node. A node learns this information implicitly by relating the token generation process for a given session to that at the adjacent nodes on the path of the session. The procedure is as follows.

A node samples every session traversing the node in round robin order. When a node samples a session in a slot, it generates a token to the session unless the number of tokens for the session at the node substantially exceeds that for the session at the adjacent nodes. This "prohibitive" difference is a window parameter W. More formally, consider the token generation process for a session i at node m. Let l and n be the nodes adjacent to node m on the path of session i (i.e., one is the immediate upstream and the other is the immediate downstream node). Let $C_{i,p}(t)$ be the number of tokens for session i at node p in slot t. When node m samples session i in slot t, then it generates a token to session i if $C_{i,m}(t) < C_{i,l}(t) + W$ and $C_{i,m}(t) < C_{i,n}(t) + W$. If one of these conditions do not hold (i.e., $C_{i,m}(t) \geq C_{i,l}(t) + W$ or $C_{i,m}(t) \geq C_{i,n}(t) + W$), session i does not receive a token in the slot, and the node samples the next session in the round robin turn. Note that the source(destination) node of a session has only one adjacent node for the session. Thus such a node takes the token generation decisions basing on the number of tokens at only one adjacent node. Tokens are never removed from a node.

It follows from the token generation process that the number of tokens for a session at two adjacent nodes on the path of the session differ by W or less at any time t, and the difference is at most LW for any two nodes on the path of a session, where L is the number of hops in the session path. Thus the token generation rate for a session is equal at any two nodes on the path of the session. Since intuitively the bottleneck node is the most congested, the sampling rate for a session is the least at its bottleneck node, and hence the token generation rate at each node is upper bounded by this sampling rate. It turns out that this sampling rate is maxmin fair, and furthermore the token generation rate of a session exactly equals the sampling rate at its bottleneck node. Thus the token generation rate is maxmin fair for each session as well. Lemma 3 states the precise analytical result.

Lemma 3: Let $r_1, r_2, \ldots r_N$ be the maxmin fair rates of the sessions. Let $C_{i,n}(t)$ be the number of tokens for session i at node n at time t. Let $W \geq W_0$, where W_0 is a constant whose value depends on the system parameters. Then in any interval (x, y)

$$|C_{i,n}(x) - C_{i,n}(y) - r_i(y - x)| \leq \varrho$$

Here, ϱ is a constant whose value does not depend on the choice of the interval (x, y), and depends only on the topology.

The interpretation of this lemma is that the number of tokens generated in any interval differs from the maxmin fair number by at most a constant. We present expressions for ϱ, W_0 in section VI, where we prove this lemma.

This concept of implicitly discovering the bottleneck information by considering the bandwidth allocation process at neighboring nodes is commonly termed as "back-pressure" and has been used for fair bandwidth allocation in wireline case previously[5]. However, the procedures differ significantly in the two cases as the bandwidth allocation constraints are inherently different.

Whenever the source node of a session receives a new token, it releases a new packet. Thus the packet release process is maxmin fair. The scheduling strategy transmits the released packets along the pre-specified routes to the desired destinations. We now describe the maximum backlog based scheduling[18] which is used for this purpose. Maximum backlog based scheduling assigns a weight to each link in a specific manner and thereafter schedules the links which constitute a maximum weighted matching. The weight is assigned as follows. The difference in backlog of a session in a link is equal to the difference between the number of packets of the session waiting at the source node of the link and that at the destination node of the link. The weight of a link is the maximum difference in backlog of a session in the link. This weight computation procedure considers only the released packets. The distinction between the released packets and the packets which have not been released is necessary only at the source nodes as only the released packets reach the intermediate nodes. When a link is scheduled, a packet of the session with the maximum difference in backlog in the link is served, i.e., transferred from the link source to the link destination. It has been shown that the maximum backlog based scheduling stabilizes a network as long as the packet arrival process is feasible[18]. The packet arrival process in the current network is the packet release process. Packets are released for each session at the maxmin fair rate (Lemma 3) which is feasible by definition. Thus the stability result indicates that the system attains maxmin fair bandwidth. This result is formally stated in Theorem 1.

Theorem 1: Let $r_1, r_2, \ldots r_N$ be the maxmin fair rates of the sessions. Let $D_i(t)$ be the number of packets for session i which have reached destination by time t. Then if $W \geq W_0$ in any interval (x, y),

$$|D_i(y) - D_i(x) - r_i(y - x)| \leq \kappa$$

Here, κ is a constant whose value does not depend on the size of the interval (x, y), and depends only on the topology.

Proof of Theorem 1: If maxmin fair rate of session i is r_i, then from Lemma 3 the number of packets released from the source of session i in any interval (x, y) differs from $r_i(y - x)$ by at most ϱ, where ϱ is a constant whose value does not depend on the interval (x, y). Since $(r_1, \ldots, r_N)$ is feasible, results in [17] show that the packet queue lengths are bounded in every node of a wireless network, if the maximum difference backlog based scheduling is used. It follows that the number of session i packets which reach the destination of session i in any interval (x, y) differs from $r_i(y - x)$ by at most a constant κ, where κ is a constant whose value does not depend on the interval (x, y). The value of κ depends on system constants and ϱ. □

Theorem 1 shows that in any interval the total number of packets of a session delivered to the destination differs from the maxmin fair number by at most a constant. It follows that the long term rates are maxmin fair.

B. Generalization for addressing the unsaturated case

We mention the necessary modifications to address the case when the system is not saturated i.e., source nodes do not always have packets for transmission. Only the packet release process and the token generation at the source node need to be altered. The key difference is that the source node of a session may not be able to release a new packet for transmission whenever it generates a new token for the session, since a new packet may not be available at that time. So the source node stores such tokens for transmission of future packets, and also distinguishes between the tokens which have been used for packet release ("used tokens") and those which are stored for future release ("unused tokens"). When a session packet is generated at the source node, it is released for transmission if there is an outstanding unused token, and the status of the corresponding token becomes "used." If there is no unused token, then the packet waits for a new token generation. If a source has W unused tokens for a session, then the token generation process does not release a new token for the session and samples the next session in the sampling order. The rest remains the same as before. Also, the node immediately downstream of the source node considers the total number of tokens for the session at the source node while deciding whether or not to generate a token for the session, and does not distinguish between the used and the unused tokens for this purpose. The modification has been presented in Figure 2.

The restriction on the number of unused tokens at the source node adapts the bandwidth allocation in accordance with the packet generation rates. If a session has low packet generation rate, then the system generates fewer tokens for the session at the source because of this restriction. This means that the session receives fewer tokens at other nodes as well, since back-pressure upper bounds the difference between the number of tokens for a session in

Procedure Token Generation at Source Node(source i)
begin
Let node n be source of session i.
Sample each session traversing node n in round robin order.
Token generation procedure is similar for all sessions other than i.
Let node l be immediately downstream of node n in the path of session i.
When session i is sampled in slot t:
if $C_{i,n}(t) < C_{i,l}(t) + W$ and $C_{i,n}^{\text{unused}}(t) < W$ **then**
Generate a token for session i in slot t ($C_{i,n}(t+1) = C_{i,n}(t) + 1$);

if no session i packet is waiting for transmission at node n, **then**
Increment the number of unused tokens of session i
($C_{i,n}^{\text{unused}}(t+1) = C_{i,n}^{\text{unused}}(t) + 1$)
else
Release a session i packet for transmission
else
Do not generate token for session i ($C_{i,n}(t+1) = C_{i,n}(t)$) and
Sample the next session in the round robin order
end

Procedure Handling Packet Generation(source i)
begin
When a new packet is generated at the source n of session i,

if there is an unused token for session i ($C_{i,n}^{\text{unused}}(t) > 0$) **then**
Release the new packet for transmission
Decrement the number of unused tokens
($C_{i,n}^{\text{unused}}(t+1) = C_{i,n}^{\text{unused}}(t) - 1$)
else
Store the new packet for future release
end

Fig. 2. The figure shows the pseudo code of the fair bandwidth allocation algorithm for systems with un-saturated sessions. Only the token generation process at the source nodes is shown. The token generation process at an intermediate node and the packet scheduling policy is the same as the basic algorithm and omitted here for brevity.

any two nodes by a constant LW. Tokens are like transmission opportunities. Thus the sessions with less packet generation rate obtain less bandwidth as required by the definition of maxmin fairness (page 2).

The operation of the algorithm does not assume any specific arrival process. However, the analytical service guarantees presented in Lemma 3 and Theorem 1 hold for pseudo-deterministic(ρ, σ) arrival processes, i.e., if the total number of packets arriving for any session i in any interval of length t is upper bounded by $t\rho + \sigma$ and lower bounded by $t\rho - \sigma$ for arbitrary t. Here, ρ is the long term arrival rate of a session and σ is the burstiness. The proofs are similar to that in the saturated case and have been omitted for brevity.

C. Generalization for attaining weighted maxmin fairness

We describe the alterations needed in the basic algorithm to accommodate the weighted maxmin fair case. We describe the procedure for the saturated case only. The unsaturated case can be captured using the technique in the previous subsection. Only the token generation process of the basic algorithm need to be altered. We first describe the token generation in this case.

Procedure Token Generation(node m)
begin
Sample the sessions traversing node m which has $\min_i C_{i,m}(t)/W_i$.
Let session i traverse node m and
nodes l, n be adjacent to node m in the path of the session
When session i is sampled in slot t:
if $C_{i,m}(t) < C_{i,l}(t) + W$ and $C_{i,m}(t) < C_{i,n}(t) + W$ **then**
Generate a token for session i in slot t ($C_{i,m}(t+1) = C_{i,m}(t) + 1$);
else
Do not generate token for session i, ($C_{i,m}(t+1) = C_{i,m}(t)$) and
Sample the session with the next larger value of $C_{j,m}(t)/W_j$.
end

Fig. 3. Token generation procedure for attaining weighted maxmin fairness. The packet scheduling and the packet release policy is the same as the basic algorithm.

A node samples the sessions in accordance with the weights and the number of accumulated tokens. Let weight of session i be G_i. Node n samples the session which has the minimum weighted number of tokens, i.e, minimum value of $C_{i,m}(t)/G_i$ among all sessions i traversing the node. When a node samples a session, it decides whether or not to generate a token for the session based on the number of tokens for the session at the adjacent nodes. This generation condition is the same as that for the basic algorithm. The packet release and the scheduling are the same as that for the basic case as well. We have described the token generation procedure in Figure 3.

We now present the intuition behind the changes. The important distinction between the unweighted and the weighted maxmin fairness is that in the former the system attempts to provide equal bandwidth to all sessions and the differences in bandwidth allocation are due to congestion only. In the latter, the differences in the bandwidth allocation are due to the distinction in weights as well as congestion. Thus the basic algorithm samples the sessions in round robin order and uses back-pressure to detect congestion and divert the unused resources towards the sessions which face less congestion. For weighted maxmin fairness, the sampling reflects the weights so that the session with higher weights are sampled more often. Thereafter, back-pressure is used to improve the bandwidth allocation of less congested sessions without reducing the bandwidth allocation of the more congested ones.

Theorem 1 and Lemma 3 hold with $r_1, r_2, \ldots r_N$ being the weighted maxmin fair rates. The proofs follow similar reasoning as that for the unweighted maxmin fair case.

IV. Performance Evaluation

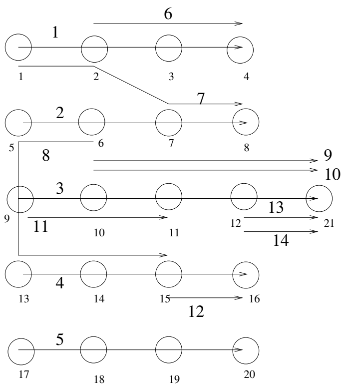

Fig. 4. The figure shows a sample topology with 21 nodes and 14 sessions which is used for performance evaluation of the fairness algorithms.

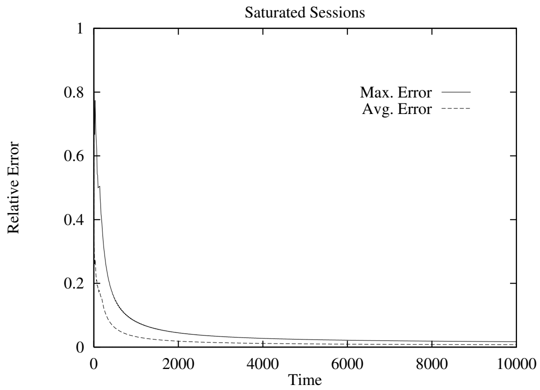

Fig. 5. The figure plots the maximum and average relative difference between the token generation rate and the maxmin fair rates for the system shown in Figure 4 with $W = 5$. All sessions are saturated in this case. The maximum and average errors are computed over all sessions at any time t.

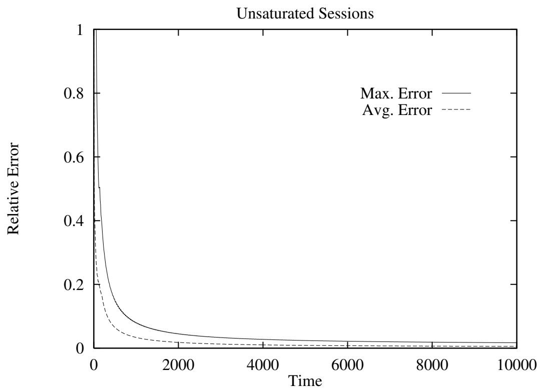

Fig. 6. The figure plots the maximum and average relative difference between the packet release rate and the maxmin fair rates for the system shown in Figure 4 with $W = 5$. Session 7 receives packet at rate 0.1 per unit time, while other sessions are saturated. The maximum and average errors are computed over all sessions at any time t.

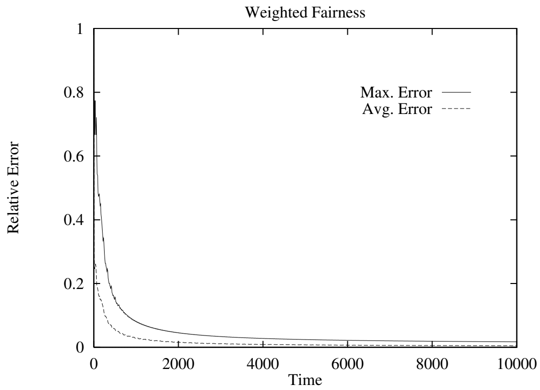

Fig. 7. The figure plots the maximum and average relative difference between the token generation rate and the weighted maxmin fair rates for the system shown in Figure 4 with $W = 5$. Session 7 has weight 2 while all other sessions have weight 1. The maximum and average errors are computed over all sessions at any time t.

We now evaluate the performance of the fairness algorithms presented in the previous section by simulation. The objective of the simulation has been two-fold: (a)to examine the time required for convergence to the maxmin fair rates and (b) the impact of the choice of the window parameter on the convergence result. For experimental evaluation we focus on the token generation procedure only. This is because the other main component of the framework, i.e., the maximum backlog based scheduling[18] has been known to attain any feasible rate as long as the packet arrival process is feasible. We know from the analytical results in the previous section that the total number of tokens generated and hence the number of packets released differs from the maxmin fair number by at most a constant in any interval. The analysis in section VI presents an upper bound for this constant through iterations (5) to (12). However, the upper bounds seems to be a pessimistic one. We thus use simulation to examine the time required for the token generation rate (and hence the packet release rate) to converge to the fair rates.

Execution of the algorithm does not require any knowledge of the network topology and traffic characteristics. However, the analytical results are guaranteed for sufficiently large values of the window parameter W, i.e., if W satisfies a lower bound which can be computed through iterations (5) to (12). This lower bound depends on the system parameters like the number of sessions, the length of session paths and arrival parameters (for the unsaturated case) etc. The expression for the bound is overly pessimistic ($W > 2^{NL}$ is necessary) and in general difficult to compute without explicit knowledge of the network topology. Besides window size is directly related to the packet queue lengths at the nodes and this affects delay and buffer sizes. Thus it is important that small window sizes be adequate for our objectives. An important motivation behind the simulation study has been to investigate the sensitivity of the convergence towards the choice of the window parameter W.

We present simulation results for a network of 21 nodes and 14 sessions as shows in Figure 4. We choose $W = 5$ all through. We first consider un-weighted maxmin fairness and consider two cases separately: (a) all sessions are saturated (i.e., all sources generate packet at a high rate and always have packets ready for transmission) and (b) the systems consist of both saturated and unsaturated sessions (the latter generate packets at a given rate and may not always have packets ready for transmission at the source).

The basic algorithm presented in Figure 1 applies in the saturated case. The maxmin fair bandwidth allocation is $(0.33, 0.67, 0.2, 0.33, 1, 0.33, 0.33, 0.33, 0.2, 0.2, 0.4, 0.33, 0.2, 0.2)$. The number of tokens generated for any session in its source node equals the number of packets released for transmission at the source node in this case. We measure the long term token generation rate for each session i at its source as a function of time t, i.e., if node n is the source of session i, we measure $C_{i,n}(t)/t$ at every t. Subsequently we consider the relative error between this value and the maxmin fair rate ($|1 - \frac{C_{i,n}(t)}{r_i t}|$) for each session i. We plot the maximum and average relative errors taken over all sessions at each time t in Figure 5.

Now we consider the case when session 7 is not saturated and generates packets at rate 0.1. The new demand constraint (page 2) of 0.1 for session 7 changes the overall maxmin fair bandwidth allocation to $(0.45, 0.67, 0.2, 0.33, 1, 0.45, 0.1, 0.33, 0.2, 0.2, 0.4, 0.33, 0.2, 0.2)$. Bandwidth allocation for session 7 decreases to 0.1 on account of the demand constraint. Consequently, the system increases the allocations for sessions 1 and 6 which share some links with session 7. The algorithm in Figure 2 handles this general case where some sessions can be unsaturated. We plot the maximum and average relative errors between the packet release rates and the maxmin fair rates in Figure 6.

Finally we consider the objective of weighted maxmin fairness. The algorithm presented in Figure 3 provides the token generation procedure for this case. We assume a weight of 2 for session 7 and a weight of 1 for other sessions. This increases the maxmin fair rate of session 7 to 0.5 and decreases that of sessions $1, 2$ and 6 to $0.25, 0.5, 0.25$ respectively. The weighted maxmin fair bandwidth allocation is $(0.25, 0.5, 0.2, 0.33, 1, 0.25, 0.5, 0.33, 0.2, 0.2, 0.4, 0.33, 0.2, 0.2)$. Figure 7 shows the decay of the maximum and average relative errors of the token generation rate with respect to the weighted maxmin fair bandwidth allocation.

We make the following observation from Figures 5 to 7. The average relative error decays noticeably faster than the maximum relative error. The average relative error is less than 5% within 500 slots. The maximum relative error decays somewhat slower indicating that a few sessions experience slower convergence. The packet release rates (and hence the token generation rates) converge to the maxmin fair bandwidth allocation even though $W = 5$, while the lower bound for guaranteed convergence is 2^{70}! We simulated several other topologies and observed similar results in all of them. We thus conclude that: the packet release rate converges to the maxmin fair bandwidth rapidly on an average. Also, in practice convergence is not sensitive to the choice of W and moderate values of W in the range of 5 to 10 ensure convergence. This is a desirable feature as small window sizes can now be used to control the delay and buffer requirements.

V. Discussion and Conclusion

The focus of this paper has been to design an algorithm for attaining maxmin fairness in a multihop scenario, and establish its efficacy from a conceptual point of view. Design of a wireless protocol based on this framework remains an interesting topic for future research. Nevertheless, we describe certain salient features which closely relate to implementation.

The fair bandwidth computation via token generation and the scheduling operate in parallel. A sequential operation of the two, e.g., starting the scheduling after completing the computation increase the overall delay in attaining the desired bandwidth allocation, which may be detrimental if the sessions are short lived. In our framework, the scheduling and the packet release process proceeds from the start, and uses the most recent output of the token generation process, irrespective of whether the token generation rate has converged to the maxmin fair rate or not. The system still stabilizes and the throughputs are the maxmin fair bandwidths.

A dynamic scenario can be accommodated where sessions can join and leave any time, as the overall scheme need not restart for any such change, and the analytical guarantees hold.

A node needs to know the number of tokens at its neighbors. This information can be periodically interchanged along with packet transmission. However, there will be certain delay involved in transmitting such information. The performance guarantees hold even when a node knows the number of tokens at its neighbors at a previous time instant, as long as the time lag is upper bounded. This happens since the number of tokens generated by a node in any upper bounded interval is also upper bounded. Thus the inaccuracy in the information is upper bounded as well, and hence the number of tokens generated for a session in any two nodes in its path is still upper bounded by a constant, where the constant depends on W, the number of hops between the nodes and the upper bound for the time lag. Thus the token generation rate for a session at any two nodes on its path is still equal, and this rate is governed by the bandwidth offered at the bottleneck node as before. Hence the guarantees still hold. In fact, the feedback delay independence property of a similar back-pressure technique has been shown in [15].

The token generation and the packet release processes are distributed, i.e., a node can execute them with the knowledge of the status of its one-hop neighbors only. However, the maximum difference backlog based scheduling is a centralized procedure. Since the execution of the token generation and the packet release processes are not tightly coupled to this particular scheduling, a distributed scheduling mechanism can also function instead. However, the performance guarantees may be weaker, but that is only expected as normally a distributed strategy attains less overall throughput than a centralized one. Future research will be directed towards the investigation of a token generation based fair bandwidth allocation scheme using an appropriate distributed scheduling strategy.

The system generates tokens, but never removes them. However, tokens are logical quantities, and only the number of tokens generated for each session need be tracked. Thus problem arises only when the number of tokens become so large that a register overflows. The performance guarantees hold if the tokens are removed without affecting the difference in the number of tokens for a session in any two nodes in its path. This calls for synchronization in the token removal process among the nodes in the path of the session, and can be attained if all such nodes use each others clock information to agree on removing a certain number of tokens at certain intervals. Since registers are normally large, tokens need not be removed for short lived sessions. The removal process can be executed by exchanging certain synchronizing information once in long intervals for long-lived sessions. This increases the system overhead only marginally.

VI. Proof of Lemma 3

The proof can be outlined as follows. The first step is to prove that if every node on the path of a session samples the session at a rate r or higher, then the session receives tokens at the rate r or higher from every node in its path (Lemma 4). Next step shows that the sampling rate equals the maxmin fair rate at every node on the path of a session (Lemma 5). The result follows.

We first introduce some terminologies. Let $S_{i,n}(t)$ denote the number of times session i is sampled at node n

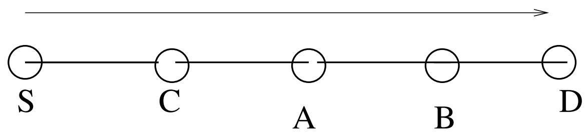

Fig. 8. We show the path of a session between source S and destination D, spanning nodes C, A, B. The $A-$set consists of S, C, A and the $B-$set consists of nodes B, D.

before time t, and L_i is the length of the path of session i. Lemma 4 can be stated as follows.

Lemma 4: Consider an arbitrary K and a sequence of K disjoint intervals, (t_l, w_l), $l = 1, \ldots, K$ which satisfies the following property. For any arbitrary M and arbitrary sequence of subintervals (x_m, y_m), $m = 1, \ldots, M$ where each (x_m, y_m) is in some (t_l, w_l) for every flow i, on every node n on its path

$$\sum_{m=1}^{M} (S_{i,n}(y_m) - S_{i,n}(x_m)) \geq r \sum_{m=1}^{M} (y_m - x_m) - e - Mf \tag{1}$$

where e and f are some constants independent of the choice of M and the sub intervals (x_m, y_m), $m = 1, \ldots, M$. If $W > 2^{L_i - 2} f$, then, $\sum_{l=1}^{K} (C_{i,n}(w_l) - C_{i,n}(t_l)) \geq r \sum_{l=1}^{K} (w_l - t_l) - 2^{L_i - 1} e - (L_i - 1) 2^{L_i - 2} f - K 2^{L_i - 1} f$ on every node n on the path of a session i.

Lemma 4 states that if a session is sampled approximately $r(y - x)$ times in every node in its path in every interval (x, y), then it receives approximately $r(y - x)$ tokens from every node in its path in every interval (x, y).

We discuss the proof here. We present the precise argument in the next section. We allow flows which span only one node, in addition to the flows with multiple nodes for mathematical convenience. This is not necessary for the result to hold, and in reality the system will consist of flows with two or more nodes only. The object of introducing this feature is to elaborate on how the token generation process would handle one node flows. Any such flow receives tokens every time it is sampled since there is no adjacent node which applies back-pressure. Lemma 4 uses an inductive argument. The induction is on the length of a flow (the number of nodes on the path of a session). The flows with one node form the base case. From the previous argument, the lemma clearly holds for this base case. (A flow spanning 2 nodes can also be taken as the base case, but that unnecessarily lengthens the proof). The induction argument is to assume that the lemma holds for all flows of length p, and then prove it for $p + 1$ (note that the system is not assumed to contain flows of all lengths). Consider a flow of length $p+1$ and two nodes A and B (Figure 8). The basic idea is that node B does not prevent any token generation at node A unless the number of tokens at A is W more than that at B. When this happens (i.e., the number of tokens at A is W more than that at B), node A does not prevent any token generation to B. Thus the nodes from B to the destination generate tokens as though they constitute a session of lower length oblivious to the presence of the nodes from A to the source. The sampling rates in these nodes satisfy the condition of the lemma (by the lemma hypothesis). By induction hypothesis, the session receives tokens at rate r at node B in these intervals. In all these intervals, number of tokens at A exceed that at B by W. Thus the token generation at A can be lower bounded by that at B, and hence satisfies the desired guarantees. In other intervals, node B does not prevent the token generation at A. Thus the nodes from the source to A behave like a session of lower length, and by induction hypothesis and the assumption on the sampling rate, node A generates tokens at the desired rate for the flow. The argument is presented more precisely in the next section.

We now state and sketch the proof of Lemma 5 which shows that a session i is sampled approximately $r_i(y-x)$ times in every node in its path in every interval (x, y), where r_i is its maxmin fair rate. Using this fact, Lemma 5 also shows an upper and lower bound on the token generation rate for each session. Let L be the maximum number of nodes on the path of any session ($L = \max_i L_i$), and β, γ, are constants which will be formulated later.

Lemma 5: Let $r_1, \ldots r_N$ be the maxmin fair rates of session $1, \ldots, N$. Consider any positive integer K, and an arbitrary non-decreasing sequence of times $x_1, y_1, \ldots, x_K, y_K$. Let $W \geq 2^{L-2}\varepsilon_1(F)$, where $\varepsilon_1(F)$ is defined later in (5) to (10). For each node A and for each session i traversing A

$$\sum_{k=1}^{K} S_{i,A}(y_k) - S_{i,A}(x_k) \geq r_i \sum_{k=1}^{K}(y_k - x_k) - \beta - K\gamma \quad (2)$$

$$\sum_{k=1}^{K} C_{i,A}(y_k) - C_{i,A}(x_k) \geq r_i \sum_{k=1}^{K}(y_k - x_k) - \beta - K\gamma \quad (3)$$

$$\sum_{k=1}^{K} C_{i,A}(y_k) - C_{i,A}(x_k) \leq r_i \sum_{k=1}^{K}(y_k - x_k) - \beta - K\gamma \quad (4)$$

Here, β, γ are constants which do not depend on the choice of the intervals and depend only on the topology.

We introduce the notion of "rank" of a session for sketching the proof. A session of rank p has the pth lowest maxmin fair rate, and $\hat{r}_p$ is the pth lowest maxmin fair rate. We actually prove that all sessions of rank p and above are sampled at a rate lower bounded by $\hat{r}_p$ at every node. Token generation rate for these sessions is also lower bounded by $\hat{r}_p$. Subsequently we show that the token generation rate of every session of rank equal to p is upper bounded by $\hat{r}_p$ at every node. The lemma follows. We use induction on the rank of a session. The base case ($p = 1$) is argued as follows. Since the sampling is round robin it can be shown that all sessions are sampled at a rate $\hat{r}_1$ or higher at every node. The lower bound on the token generation rate follows from Lemma 4. Next we show that the token generation rate is upper bounded by $\hat{r}_1$ for all sessions with rank 1. This follows because the sampling and hence the token generation rate is upper bounded by $\hat{r}_1$ at the bottleneck node, and the token generation rates are the same for the same session at different nodes on the path of the session, on account of back-pressure. For the induction case, the lower bound on the sampling rate follows since the token generation rates of sessions with rank lower than p are upper bounded by their respective maxmin fair rates, and these maxmin fair rates are lower than $\hat{r}_p$. Round robin sampling in the intervals not utilized by sessions of lower ranks allows a sampling rate of at least $\hat{r}_p$ to all other sessions. Again, the lower bound on the token generation rate follows from Lemma 4. The upper bound on the token generation rate for sessions with rank p can be proved as in the base case. The proof has been presented in Section VIII. Now we present iterations defining $\varsigma_i(j), \varepsilon_i(j), \beta, \gamma$.

$$\begin{aligned}
\varsigma_1(1) &= 0 && (5)\\
\varepsilon_1(1) &= 1 && (6)\\
\varsigma_2(p) &= 2^L\varsigma_1(p) + (L-1)2^{L-2}\varepsilon_1(p) && (7)\\
\varepsilon_2(p) &= 2^{L-1}\varepsilon_1(p) && (8)\\
\varsigma_3(p) &= d_{\max}\left(\varsigma_2(p) + \varepsilon_2(p)\right) + 2LW && (9)\\
\varepsilon_3(p) &= \varepsilon_2(p) && (10)\\
\varsigma_1(p+1) &= (d_{\max} - 1)\varsigma_3(p) && (11)\\
\varepsilon_1(p+1) &= (d_{\max} - 1)\varepsilon_3(p) + 1 && (12)
\end{aligned}$$

Also, $\beta = \varsigma_3(F)$ and $\gamma = \varepsilon_3(F)$, where F is the number of distinct ranks.

Proof of Lemma 3: Lemma 3 follows from (3) and (4) of Lemma 5 with $\varrho = \varsigma_3(F) + \varepsilon_3(F)$ and $W_0 = 2^{L-2}\varepsilon_1(F)$, where F is the number of distinct ranks and $\varsigma_3(F), \varepsilon_1(F), \varepsilon_3(F)$ have been defined via the sequence of iterations (5) to (10). □

VII. Proof of Lemma 4

Proof of Lemma 4: We assume the conditions of the lemma and subsequently prove by induction. The result holds for all sessions with only one node from the assumption on the sampling rate (condition (1)). The induction hypothesis is that the result holds for sessions with p or

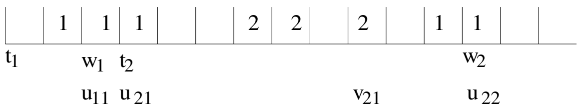

Fig. 9. We show two intervals $[t_1, w_1]$ and $[t_2, w_2]$, and some type one and two slots there marked by "1" and "2" respectively. We also show the corresponding u and v slots. Note that if the slots without numbers are ignored, then the last type one(two) slot in a sequence of type one(two) slots in a $[t_i, w_i]$ interval is a "u"("v") slot. Here $[t_1, u_{11}]$, $[t_2, u_{21}]$, $(v_{21}, u_{22}]$ are example sub-intervals which end in $u-$slots and start from the nearest $v-$slot or the start of the interval whichever is the nearest.

fewer nodes. We will show that the result holds for sessions with $p+1$ nodes. Consider any node A in the path of a session i with $p+1$ nodes. If the number of tokens of session i at A does not exceed that of its previous or next node by W or more in the intervals (t_l, w_l), $l = 1, \ldots, K$, then node A generates a token for session i every time it samples session i. Hence the result holds from condition (1). Thus we assume that there exists a node B which is adjacent to A such that $C_{i,A}(t) = C_{i,B}(t) + W$ at some time t in these intervals. We consider the slots t in these intervals where either $C_{i,A}(t)$ or $C_{i,B}(t)$ exceed the other by W. Let a slot t where $C_{i,A}(t)$ exceeds $C_{i,B}(t)$ by W be a type 1 slot, and a slot where the reverse occurs be a type 2 slots (a slot may be neither type 1 nor type 2). Thus we have sequences of type one and two slots (represented by 1s and 2s respectively), with slots which are of neither type in between(Figure 9). We would name some specific type one slots as "u" slots and some specific type two slot as "v" slots. For this ignore the slots without a number and consider each $[t_i, w_i]$ interval separately. The last type one slot in a sequence of type one slots is denoted a "u" slot, and the last type 2 slot in a sequence of type two slots is a "v" slot. The u and v slots are numbered sequentially, i.e., the mth "u-slot" of the lth interval is u_{lm}(Figure 9). Consider node A. Note that

$$
\begin{aligned}
C_{i,A}(u_{lm}) &= C_{i,B}(u_{lm}) + W && (13)\\
C_{i,B}(v_{lm}) &= C_{i,A}(v_{lm}) + W && (14)\\
C_{i,A}(t) &\leq C_{i,B}(t) + W, \forall t && (15)\\
C_{i,B}(t) &\leq C_{i,A}(t) + W, \forall t && (16)
\end{aligned}
$$

We introduce a terminology now. If the link connecting nodes A and B is removed then the set of nodes which remain connected to $A(B)$ will be referred to as the $A-$set($B-$set) (Figure 8). Let $A-$set($B-$set) consist of $a(b)$ nodes. Note that $a + b = p + 1$, $a > 0$, $b > 0$. Thus, $a \leq p$, $b \leq p$. Consider the sub-intervals ending at the u slots starting from the beginning of an interval (a t_j) (inclusive) or starting from a $v-$slot (not inclusive), whichever is the nearest (Figure 9). These sub-intervals do not consist of any type 2 slots (slots where $C_{i,B}(t) = C_{i,A}(t) + W$). Thus, node A does not prevent any session i token generation to node B in these subintervals, and thus the token generation for session i in the nodes in the $B-$set resembles that of a session of length b. Let there be I_1 such sub-intervals. Condition (1) holds for session i in every node in the $B-$ set for every set of subintervals of these I_1 sub-intervals, since any such subinterval is in $[t_j, w_j]$ for some j. Thus the total number of tokens generated for session i in these I_1 subintervals in each node of the $B-$set can be lower bounded by the induction hypothesis. If $v_{l1} > u_{l1}$ as in the example scenario depicted in Figure 9, then $(v_{lm-1}, u_{lm}]$, $m > 1$, and $[t_l, u_{l1}]$ constitute these I_1 sub-intervals, otherwise $(v_{lm}, u_{lm}]$ are these sub-intervals in $[t_l, w_l]$. For simplicity we assume that $v_{l1} > u_{l1}$ for all l (the argument remains similar if $v_{l1} < u_{l1}$ for some or all l). It follows from induction hypothesis that

$$
\begin{aligned}
&\sum_{l=1}^{K} ((C_{i,B}(u_{l1}) - C_{i,B}(t_l))\\
&+ \sum_{m=2} (C_{i,B}(u_{lm}) - C_{i,B}(v_{lm-1})))\\
&\geq r \sum_{l=1}^{K} \left((u_{l1} - t_l) + \sum_{m=2} (u_{lm} - v_{lm-1}) \right)\\
&-2^{p-1}e - (p-1)2^{p-2}f - I_1 2^{p-1} f \qquad (17)
\end{aligned}
$$

From (13) and (15)

$$
\begin{aligned}
C_{i,A}(u_{l1}) - C_{i,A}(t_l) &\geq C_{i,B}(u_{l1}) + W - C_{i,B}(t_l) - W\\
&= C_{i,B}(u_{l1}) - C_{i,B}(t_l) \qquad (18)
\end{aligned}
$$

From (13) and (14)

$$
C_{i,A}(u_{lm}) - C_{i,A}(v_{lm-1}) = C_{i,B}(u_{lm}) - C_{i,B}(v_{lm-1}) + 2W \qquad (19)
$$

From (18) and (19),

$$
\begin{aligned}
&\sum_{l=1}^{K} ((C_{i,A}(u_{l1}) - C_{i,A}(t_l))\\
&\quad + \sum_{m=2} (C_{i,A}(u_{lm}) - C_{i,A}(v_{lm-1})))\\
\geq &\sum_{l=1}^{K} ((C_{i,B}(u_{l1}) - C_{i,B}(t_l))\\
&\quad + \sum_{m=2} (C_{i,B}(u_{lm}) - C_{i,B}(v_{lm-1}))) + 2W(I_1 - K)\\
\geq &\; r \sum_{l=1}^{K} \left((u_{l1} - t_l) + \sum_{m=2} (u_{lm} - v_{lm-1}) \right)\\
&-2^{p-1}e - (p-1)2^{p-2}f - K 2^{p-1} f\\
&+(I_1 - K)(2W - 2^{p-1} f) \text{ (from (17))} \qquad (20)
\end{aligned}
$$

Now remove these I_1 intervals from $\cup_{l=1}^{K}(t_l, w_l)$, and consider the resulting intervals. Note that these resulting intervals do not contain any type 1 slots (slots where $C_{i,A}(t) = C_{i,B}(t) + W$). Thus, node B does not prevent any session i token generation to node A and thus the session i token generation in the nodes in $A-$set resembles that of a session of length a, $a \le p$. From the induction hypothesis, one can lower bound the number of session i tokens generated in node A in these intervals. Note that there are at most $I_1 + 1$ sub-intervals. These are of the form $(u_{lm}, v_{lm}]$ and $(v_{lm}, w_l]$ (if $v_{l1} > u_{l1}$). Thus,

$$\begin{aligned} & \sum_{l=1}^{K}((C_{i,A}(w_l) - C_{i,A}(v_{lm})) \\ & + \sum_{m=1}(C_{i,A}(v_{lm}) - C_{i,A}(u_{lm}))) \\ \ge \quad & r\sum_{l=1}^{K}\left((w_l - v_{lm}) + \sum_{m=1}(v_{lm} - u_{lm})\right) \\ & -2^{p-1}e - (p-1)2^{p-2}f - (I_1+1)2^{p-1}f \quad (21) \end{aligned}$$

Adding (20) and (21),

$$\begin{aligned} \sum_{l=1}^{K}(C_{i,A}(w_l) - C_{i,A}(t_l)) \quad \ge \quad & r\sum_{l=1}^{K}(w_l - t_l) - 2^p e \\ & -p2^{p-1}f - K2^p f \\ & +(I_1 - K)(2W - 2^p f) \quad (22) \end{aligned}$$

The implicit assumption here was that there exists at least one type one slot in each interval (t_l, w_l) and that motivates the summation from $l = 1$ to K in relation (17). Under this assumption $I_1 \ge K$. Note that $p + 1 \le L_i$ and thus $W > 2^{L_1 - 2}$. Thus, $W > 2^{p-1}f$. Hence, the induction hypothesis holds for session i at node A. In general, there may not be a type one slot in $[t_l, w_l]$ for all l. In this case, the summation in (17) must be over those intervals $[t_l, w_l]$ which have at least one type one slot. Let K_1 be the number of such intervals. The corresponding analysis replaces $(I_1 - K)$ by $(I_1 - K_1)$, Note that $I_1 \ge K_1$. Thus the induction hypothesis holds at all nodes for all sessions with $p + 1$ nodes on their path. □

VIII. Proof of Lemma 5

Let $d_{\max}$ be the maximum number of flows traversing a node. We will prove the following for $p = 1, \ldots, F$, where F is the total number of distinct ranks.

For each node A and for each session i traversing A and rank(i) $\ge$ p, for any positive integer K, and for any non-decreasing sequence of times $x_1, y_1, \ldots, x_K, y_K$

$$\sum_{k=1}^{K} S_{i,A}(y_k) - S_{i,A}(x_k) \ge \hat{r}_p \sum_{k=1}^{K}(y_k - x_k) - \varsigma_1(p) - K\varepsilon_1(p) \quad (23)$$

For each node A and for each session i traversing A and rank(i) $\ge$ p, for any positive integer K, and for any non-decreasing sequence of times $x_1, y_1, \ldots, x_K, y_K$

$$\sum_{k=1}^{K} C_{i,A}(y_k) - C_{i,A}(x_k) \ge \hat{r}_p \sum_{k=1}^{K}(y_k - x_k) - \varsigma_2(p) - K\varepsilon_2(p) \quad (24)$$

For each node A and for each session i traversing A and rank(i) = p, for any positive integer K, and for any non-decreasing sequence of times $x_1, y_1, \ldots, x_K, y_K$

$$\sum_{k=1}^{K} C_{i,A}(y_k) - C_{i,A}(x_k) \le \hat{r}_p \sum_{k=1}^{K}(y_k - x_k) + \varsigma_3(p) + K\varepsilon_3(p) \quad (25)$$

We prove these by induction on p. We first prove (23), (24) and (25) for $p = 1$.

Note that $\hat{r}_1 = \alpha/d_{\max}$ (recall that α is the bandwidth utilization factor). If d_A flows traverse node A, then from round robin sampling, every flow traversing node A is sampled at least $\alpha/d_A \sum_{k=1}^{Q}(y_k - x_k) - Q$ for any arbitrary Q and arbitrary sequence of nondecreasing times $x_1, y_1, \ldots, x_Q, y_Q$. Since $d_A \le d_{\max}$, $\hat{r}_1 \le \alpha/d_A$. Thus (23) holds in this case, with $\varsigma_1(1) = 0$ and $\varepsilon_1(1) = 1$.

Since $\varepsilon_F(1) \ge \varepsilon_1(1)$, $W > 2^{L-2}$. Hence (24) follows from Lemma 4 with $\varsigma_2(1) = (L-1)2^{L-2}$ and $\varepsilon_2(1) = 2^{L-1}$.

Now we prove (25) for $p = 1$. Consider a flow i with rank 1. Let A be a node on the path of flow i. Consider any sequence of non-decreasing times $x_1, y_1, \ldots, x_K, y_K$.

$$\begin{aligned} & \sum_{k=1}^{K} C_{i,A}(y_k) - C_{i,A}(x_k) \\ = \quad & C_{i,A}(y_K) - C_{i,A}(x_1) \\ & - \sum_{k=1}^{K-1}(C_{i,A}(x_{k+1}) - C_{i,A}(y_k)) \\ \le \quad & C_{i,A}(y_K) - C_{i,A}(x_1) - \hat{r}_1 \sum_{k=1}^{K-1}(x_{k+1} - y_k) \\ & + \varsigma_2(1) + (K-1)\varepsilon_2(1) \quad (26) \end{aligned}$$

The last step follows from (24) for $p = 1$. Since flow i has rank 1, $d_{\max}$ flows traverse its bottleneck node, B. Let X be the set of flows traversing node B.

$$\begin{aligned} & C_{i,B}(y_K) - C_{i,B}(x_1) \\ \le \quad & \alpha(y_K - x_1) - \sum_{m \in X \setminus \{i\}}(C_{m,B}(y_K) - C_{m,B}(x_1)) \end{aligned}$$

$$
\begin{aligned}
&\le \alpha(y_K - x_1) - (d_{\max} - 1)(\hat{r}_1(y_K - x_1) - \varsigma_2(1) \\
&\quad -\varepsilon_2(1)) \text{ (from (24))} \\
&= \hat{r}_1(y_K - x_1) + (d_{\max} - 1)\left(\varsigma_2(1) + \varepsilon_2(1)\right) \qquad (27) \\
&\qquad \text{(since } \hat{r}_1 = \alpha/d_{\max})
\end{aligned}
$$

We know that

$$
|C_{i,A}(t) - C_{i,B}(t)| \le L_i W \forall t \qquad (28)
$$

$$
\begin{aligned}
& C_{i,A}(y_K) - C_{i,A}(x_1) \\
\le\ & C_{i,B}(y_K) - C_{i,B}(x_1) + 2L_iW \text{ (from (27))} \\
\le\ & \hat{r}_1(y_K - x_1) + (d_{\max} - 1)\left(\varsigma_2(1) + \varepsilon_2(1)\right) \\
& +2L_iW \qquad \text{(from (28))} \qquad (29)
\end{aligned}
$$

From (26) and (29),

$$
\begin{aligned}
& \sum_{k=1}^{K} C_{i,A}(y_k) - C_{i,A}(x_k) \\
\le\ & \hat{r}_1 \sum_{k=1}^{K}(y_k - x_k) + d_{\max}\left(\varsigma_2(1) + \varepsilon_2(1)\right) \\
& +2LW + K\varepsilon_2(1) \qquad (30)
\end{aligned}
$$

Thus (25) follows from (30) with $\varsigma_3(1) = d_{\max}\left(\varsigma_2(1) + \varepsilon_2(1)\right) + 2LW$ and $\varepsilon_3(1) = \varepsilon_2(1)$. Thus the base case holds.

Now we assume that (23), (24) and (25) hold for rank $1, \ldots, p$, and show that these hold for $p + 1$.

We first prove (23). Consider a flow i of rank $p + 1$. Consider a node A on the path of flow i. Let $Y = \{w : w \text{ traverses } A, \text{rank}(w) \le p\}$ and $Z = \{w : w \text{ traverses } A, \text{rank}(w) \ge p + 1\}$. Now, Z includes i.

$$
\begin{aligned}
& S_{i,A}(y_k) - S_{i,A}(x_k) \\
\ge\ & \max_{w \in Z}\left(S_{i,w}(y_k) - S_{i,w}(x_k)\right) - 1 \\
& \text{(property of round robin scheduling)} \\
\ge\ & \frac{1}{|Z|} \sum_{w \in Z}\left(S_{i,w}(y_k) - S_{i,w}(x_k)\right) - 1 \\
\ge\ & \frac{\alpha(y_k - x_k) - 1 - \sum_{w \in Y}\left(C_{i,w}(y_k) - C_{i,w}(x_k)\right)}{|Z|}
\end{aligned}
$$

The chances not used by sessions in Y are offered to sessions in Z.

$$
\begin{aligned}
& \sum_{k=1}^{K}\left(S_{i,A}(y_k) - S_{i,A}(x_k)\right) \\
\ge\ & \frac{1}{Z}(\alpha \sum_{k=1}^{K}(y_k - x_k) - K \\
& - \sum_{w \in Y} \sum_{k=1}^{K}\left(C_{i,w}(y_k) - C_{i,w}(x_k)\right)) \\
\ge\ & \frac{(\alpha - \sum_{w \in Y} r_w)\sum_{k=1}^{K}(y_k - x_k)}{|Z|} \\
& - \frac{|Y|}{|Z|}\varsigma_3(p) - K\frac{1 + |Y|\varepsilon_3(p)}{|Z|}
\end{aligned}
$$

The last inequality follows from (25) by induction hypothesis and since $\text{rank}(w) \le p$ for $w \in Y$, $r_w = \hat{r}_{\text{rank}(w)}$, $\varsigma_3(j) \ge \varsigma_3(j-1)$, $\varepsilon_3(j) \ge \varepsilon_3(j-1) \forall j$. Thus,

$$
\begin{aligned}
& \sum_{k=1}^{K}\left(S_{i,A}(y_k) - S_{i,A}(x_k)\right) \\
\ge\ & \frac{\sum_{w \in Z} r_w \sum_{k=1}^{K}(y_k - x_k)}{|Z|} - \frac{|Y|}{|Z|}\varsigma_3(p) \\
& -K\frac{1 + |Y|\varepsilon_3(p)}{|Z|} \\
& \text{(since } \alpha \ge \sum_{w \in Z} r_w + \sum_{w \in Y} r_w) \\
\ge\ & \min_{w \in Z} r_w \sum_{k=1}^{K}(y_k - x_k) - \frac{|Y|}{|Z|}\varsigma_3(p) \\
& -K\frac{1 + |Y|\varepsilon_3(p)}{|Z|} \\
\ge\ & r_{p+1} \sum_{k=1}^{K}(y_k - x_k) - \frac{|Y|}{|Z|}\varsigma_3(p) \\
& -K\frac{1 + |Y|\varepsilon_3(p)}{|Z|} \qquad (31)
\end{aligned}
$$

The last inequality follows since $\text{rank}(w) \ge p + 1$, for $w \in Z$.

From (31), (23) holds for $p+1$, with $\varsigma_1(p+1) = (d_{\max} - 1)\varsigma_3(p)$, and $\varepsilon_1(p + 1) = (d_{\max} - 1)\varepsilon_3(p) + 1$.

Since $W > 2^{L-2}\varepsilon_1(p+1)$, (24) follows from Lemma 4 with $\varsigma_2(p+1) = 2^L\varsigma_1(p) + (L-1)2^{L-2}\varepsilon_1(p)$ and $\varepsilon_2(p + 1) = 2^{L-1}\varepsilon_1(p + 1)$.

Now we prove (25) for $p + 1$. The argument is similar to that for the base case. We point out the additional differences. Consider a flow i with rank $p + 1$. Let A be a node on the path of flow i. Consider any sequence of non-decreasing times $x_1, y_1, \ldots, x_K, y_K$.

$$
\begin{aligned}
& \sum_{k=1}^{K} C_{i,A}(y_k) - C_{i,A}(x_k) \\
=\ & C_{i,A}(y_K) - C_{i,A}(x_1) \\
& - \sum_{k=1}^{K-1} C_{i,A}(x_{k+1}) - C_{i,A}(y_k) \\
\le\ & C_{i,A}(y_K) - C_{i,A}(x_1) - \hat{r}_{p+1} \sum_{k=1}^{K-1}(x_{k+1} - y_k) \\
& +\varsigma_2(p + 1) + (K - 1)\varepsilon_2(p + 1) \qquad (32)
\end{aligned}
$$

The last step follows from (24) for $p+1$. Since flow i has rank $p+1$, all flows traversing its bottleneck node B must have rank less than or equal to $p+1$. Let X be the set of flows traversing node B. Let the number of flows traversing node B be d_B.

$$\begin{aligned} & C_{i,B}(y_K) - C_{i,B}(x_1) \\ \leq\ & \alpha(y_K - x_1) - \sum_{m \in X \setminus \{i\}} (C_{m,B}(y_K) - C_{m,B}(x_1)) \\ \leq\ & \alpha(y_K - x_1) - \sum_{m \in X \setminus \{i\}} (r_m(y_K - x_1) - \varsigma_2(p+1) \\ & -\varepsilon_2(p+1)) \qquad \text{(from (24))} \\ =\ & \hat{r}_{p+1}(y_K - x_1) + (d_B - 1)(\varsigma_2(p+1) \\ & +\varepsilon_2(p+1)) \end{aligned} \tag{33}$$

From (32) and (33), following the arguments in the base case,

$$\begin{aligned} & \sum_{k=1}^{K} C_{i,A}(y_k) - C_{i,A}(x_k) \\ \leq\ & \hat{r}_{p+1} \sum_{k=1}^{K} (y_k - x_k) + d_{\max}\left(\varsigma_2(p+1) + \varepsilon_2(p+1)\right) \\ & +2LW + K\varepsilon_2(p+1) \end{aligned} \tag{34}$$

Thus (25) follows from (34) with $\varsigma_3(p+1) = d_{\max}\left(\varsigma_2(p+1) + \varepsilon_2(p+1)\right) + 2LW$ and $\varepsilon_3(p+1) = \varepsilon_2(p+1)$. Thus the induction case holds.

Note that $\varsigma_i(x), \varepsilon_i(x)$ are increasing in both i and x. From (23), (24) and (25), Lemma 5 holds with $\beta = \varsigma_3(F)$ and $\gamma = \varepsilon_3(F)$, where F is the number of distinct ranks. $\square$

References

[1] D. J. Baker and A. Ephremides. The architectural organization of a packet radio network via a distributed algorithm. *IEEE Transactions on Communications, pp. 1694-1701*, 29, 1981.

[2] D. J. Baker, J. Wieselthier, and A. Ephremides. A distributed algorithm for scheduling the activation of links in a self-organizing, mobile, radio network. *Proceedings of IEEE International Conference on Communications,1982*, June 1982.

[3] D. Bertsekas and R. Gallager. *Data Networks*. Prentice-Hall, 1987.

[4] A. Demers, S. Keshav, and S. Shenker. Analysis and simulation of a fair-queueing algorithm. *Proceedings of ACM SigComm 1989*, 1(1), 1990.

[5] E. Hahne. Round-robin scheduling for max-min fairness in data networks. *IEEE Journal on Selected Areas in Communications*, 9(7), September 1991.

[6] B. Hajek and G. Sasaki. Link scheduling in polynomial time. *IEEE Transactions on Information Theory*, 34(5), December 1988.

[7] X.L. Huang and B. Bensaou. On max-min fairness and scheduling in wireless ad-hoc networks: Analytical framework and implementation. *Proceedings of IEEE/ACM MobiHoc, 2001*, October 2001.

[8] N. Johansson, U. Korner, and L. Tassiulas. A distributed scheduling algorithm for a bluetooth scatternet. *Proceedings Of the 17th International Teletraffic Congress, 2001*, September 2001.

[9] J. Ju and V.O.K. Li. An optimal topology transparent scheduling method in multihop packet radio networks. *IEEE Transactions in Networking*, 6(3), June 1998.

[10] H. Luo, S. Lu, and V. Bharghavan. A new model for packet scheduling in multihop wireless networks. *Proceedings of ACM Mobicom, 2000*, August 2000.

[11] B. Miller and C. Bisdikian. *Bluetooth Revealed: The Insider's Guide to an Open Specification for Global Wireless Communications*. Prentice-Hall, 2000.

[12] A. Parekh and R. Gallager. A generalized processor sharing approach to flow control - the single node case. *IEEE/ACM Transactions on Networking*, 1, June 1993.

[13] M. Post, P. Sarachik, and A Kershenbaum. A biased greedy algorithm for scheduling multihop radio networks. *Proceedings of 19th Annual Conference on Information Sciences and Systems, Johns Hopkins Univ.*, March 1985.

[14] A. Racz, G. Miklos, F. Kubinszky, and A. Valko. A pseudo random coordinated scheduling algorithm for bluetooth scatternets. *Proceedings of IEEE/ACM MobiHoc, 2001*, October 2001.

[15] S. Sarkar and L. Tassiulas. Scheduling for maxmin fairness in multirate multicast networks. *Proceedings of INFOCOM, 2001*, May 2001.

[16] X. Gao T. Nandagopal, T. Kim and V. Bharghavan. Achieving mac layer fairness in wireless packet networks. *Proceedings of ACM Mobicom, 2000*, August 2000.

[17] L. Tassiulas. Adaptive back-pressure congestion control based on local information. *IEEE Transactions on Automatic Control*, 40(2), 1995.

[18] L. Tassiulas and A. Ephremidis. Stability properties of constrained queueing systems and scheduling for maximum throughput in multihop radio networks. *IEEE Transactions on Automatic Control*, 37(12), 1992.

[19] L. Tassiulas and S. Sarkar. Maxmin fair scheduling in wireless networks. *Proceedings of INFOCOM'2002*, June 2002.