

# TECHNICAL RESEARCH REPORT

Distributed Algorithms for Computation of Fair Rates in  
Multirate Multicast Trees

*by Saswati Sarkar, Leandros Tassiulas*

**T.R. 99-42**



*ISR develops, applies and teaches advanced methodologies of design and analysis to solve complex, hierarchical, heterogeneous and dynamic problems of engineering technology and systems for industry and government.*

*ISR is a permanent institute of the University of Maryland, within the Glenn L. Martin Institute of Technology/A. James Clark School of Engineering. It is a National Science Foundation Engineering Research Center.*

**Web site <http://www.isr.umd.edu>**

# Distributed Algorithms for Computation of Fair Rates in Multirate Multicast Trees

*Saswati Sarkar* and *Leandros Tassiulas*

Dept. of Electrical and Computer Engineering and Institute for Systems Research  
University of Maryland, College Park, MD, USA  
email addresses: swati@eng.umd.edu, leandros@isr.umd.edu

## Abstract

We study fairness in arbitrary networks with multicast capabilities. Multicast traffic in internet and ATM provides a motivation for studying these networks. A study of fairness in multicast networks poses several interesting problems e.g., the issue of *intra-session* fairness in addition to that of *inter-session* fairness in unicast networks. We develop a mathematical framework to model the fair allocation of bandwidth in multicast networks with minimum and maximum rate constraints. We present distributed algorithms for computation of maxmin fair rates allocated to various source-destination pairs.

## 1 Introduction

Multicasting provides an efficient way of transmitting data from a sender to a group of receivers. A single source node or a group of source nodes sends identical messages simultaneously to multiple destination nodes. Single destination or unicast and broadcast to the entire network are special cases of multicast. Multicast applications include collaborative applications like audio or video teleconferencing, video-on-demand services, distributed databases, distribution of software, financial information, electronic newspapers, billing records, medical images, weather maps and experimental data, distributed interactive simulation (DIS) activities such as tank battle simulations. Many distributed systems such as the V System[13] and the Andrew distributed computing environment[28], popular protocol suites like Sun's broadcast RPC service[23] and IBMs NetBIOS[19] are using multicasting. Multicasting has been used primarily in the Internet, but future ATM networks are likely to deploy multicasting in a large scale, particularly in applications like broadcast video, video-conferencing, multiparty telephony and

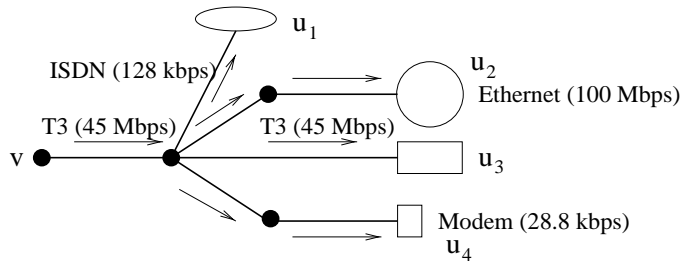


Figure 1:

workgroup applications[12]. In general many multicast sessions simultaneously share the network resources. Ideally all sessions should have a fair share of bandwidth. This issue of *inter-session* fairness have been studied extensively in unicast networks. Multicasting poses some specific challenges in this regard. This is because of network heterogeneity. A single session may have many destinations, and end systems can have widely varying bandwidth connectivities. On one hand there are fast ethernets (100 Mbps) and on the other hand there are slow modems (28.8 kbps). The paths to different destinations may have different bandwidth capacities, e.g., one may consist of multi-megabit links, such as, T3 (45 Mbps) and another may have a 128 kbps ISDN line (Refer to the network shown in figure 1 for an example). Every receiver would like to receive service at a rate commensurate with its capabilities and the capacity of the path leading to it from the source independent of the capabilities of the other receivers of the same session. This is the issue of *intra-session* fairness. Besides, like in unicast, there is the issue of *inter-session* fairness, that is fairness of members across multiple sessions. So multicasting poses the issue of *intra-session* fairness in addition to that of *inter-session* fairness.

As the Internet evolves to higher speed and larger size, the problems caused by heterogeneity will only get worse. A single rate of transmission per session is likely to either overwhelm the slow receivers or starve the fast ones, in absence of additional provisions. There are three alternative approaches, simulcast, transcoding and layered transmission. Simulcast advocates that every source maintains multiple streams carrying the same information but transmitted at different rates and quality, targetted at receivers with different capabilities[10]. Depending upon the individual capabilities, the receivers are partitioned across groups and each group subscribes to one stream. The rate

of the streams can be controlled to attain a fair share. However this is bandwidth inefficient as the same basic information is replicated across all streams. This contradicts the basic principle of multicast that messages need only be replicated at forking nodes. Besides, unless there are as many groups as the number of receivers, the problem of heterogeneity remains to a limited extent. The other two approaches are bandwidth efficient. In transcoding, the source transmits at a rate matching the fastest of its receivers. The transmission rates are transcoded at the intermediate nodes to match the capabilities of slower receivers downstream[31]. At every link, the transmission rate of a session is equal to that of the fastest session receiver downstream of the link. Video gateways are generally used for transcoding[2]. The last approach is to have a hierarchical or a layered transmission scheme. In this approach, a signal is encoded into a number of layers that can be incrementally combined to provide progressive refinement. The different layers of a multicast group are considered different multicast groups and receivers adapt to congestion by adding and dropping layers, where adding a layer is joining a multicast group and dropping a layer is leaving a group[15]. Again the number of layers of a session in a link is the maximum of the number of layers of the session receivers downstream. This layered transmission scheme have been used for both video[31] and audio[7] transmissions over the internet and has potentials for use in ATM networks as well[16]. Transcoding (layered transmission) schemes can be used to attain inter-session and intra-session fairness, in a bandwidth efficient manner, by having the receivers subscribe to a “fair” rate ( number of layers ). We assume that the network has either of these two capabilities.

We have proposed a routing and scheduling policy in [29] which stabilizes the system, if the network can accomodate all the traffic demands, without congestion. However, resource limitations may not permit this always. Fairness of resource allocation becomes important in such a scenario. The problem of fair allocation of bandwidths to multicast sessions under the constraint that all receivers of the same session must receive service at the same rate has been investigated in [32]. Intra-session fairness can not be achieved by a single rate of transmission per session on account of network heterogeneity. As [27] demonstrates formally, fairness properties of a multicast network improves if multi-rate transmission is used instead of single rate transmission, as [32] advocates. [10] advocates simulcast, but that is bandwidth inefficient. Fair allocation of layers (rates) in presence of layered transmission

(transcoding) provisions have not been cultivated very systematically, until recently[27]. There are two well known network protocols for layered transmission, RLM (Receiver-driven Layered Multicast)[25] and LVMR (Layered Video Multicast with Retransmissions)[20]. The goal of these approaches is to achieve improved intra-session fairness. However, as [21] points out, neither handles inter-session fairness very well, when there are multiple sessions competing for bandwidth. [21] proposes a scheme for fair allocation of layers for multi-session layered video multicast which strives to rectify this defect in RLM and LVMR. The authors present empirical evidence that the scheme improves inter-session fairness for networks with multiple video sessions sharing only one link. They mention that if  $M$  video streams share a link and no stream has bandwidth constraint on other links or end systems, then the layer difference between any two streams is either 0 or 1 in the steady state. But typically, streams would have bandwidth constraint on other links as well. There is no experimental or analytical evidence that the scheme works well for more complex networks, with sessions sharing several links with each other. In absence of further mechanisms, like elaborate scheduling policies, it may not be possible to establish conclusively that the scheme attains fair allocation of rates as per some well defined notion of fairness, like maxmin fairness for example. Besides [21] does not make any effort towards the computation of the actual rates or the number of layers allocated to the receivers in an arbitrary network, under some well defined notion of fairness. An algorithm for computation of maxmin fair rates in a multirate multicast network has been proposed in [27]. However, this algorithm requires global knowledge of system states for computation of the maxmin fair rates. Current networks have large sizes. Thus global knowledge of system states may not be present at any single point in the network. Distributed algorithms for computation of fair rates are more useful in this context.

We start with a mathematical formulation of the problem. We adopt the notion of maxmin fairness[6]. We would define this notion more precisely later, but informally speaking a rate allocation is maxmin fair, if no receiver can be allocated a higher rate without hurting another receiver having equal or lower rate. Maxmin fairness is a good notion of fairness and as [27] points out, maxmin fairness satisfies many intuitive fairness properties in a multirate multicast network. We present an algorithm for computation of maxmin fair rates in an arbitrary network with any number of multicast sessions. This algorithm does not require global knowledge of system states

during any stage and is thus amenable to distributed computation. We give a framework for distributed computation of maxmin fair rates. It turns out that the algorithms for computation of maxmin fair rates in unicast networks or multicast networks with a single rate of transmission per session are special cases of the algorithm we present here. However, the distributed framework based on this algorithm makes certain assumptions which may not hold always. We present another distributed algorithm for computation of maxmin fair rates which is slower than the first one in the worst case, but operates under very general assumptions. We address generic mechanism for allocation of the rates, once the rates are computed. Finally, we conclude with some future directions of research. We think a unified mathematical framework can model the issue of intra-session and inter-session fairness for multicast in both ATM and the internet. The modalities of the rate allocation and the implementational details may differ in the two scenario. As a first step, we do not distinguish between the two. Thus our algorithms are applicable in very general scenario. ATM sessions often have minimum rate requirements and maximum rate constraints. These parameters are negotiated during connection establishment phase. Our model is general enough to accomodate these requirements. Internet sessions do not have these requirements because there is no connection establishment phase. The minimum rate requirements have not been incorporated in any existing model for multirate multicast networks.

This report is organized as follows. Section 2 describes the problem of maxmin fairness for multicast transmission and presents the mathematical framework used to model the problem. This section also presents an interesting property of the maxmin fair rate allocation. Section 3.1 presents an algorithm for computation of the maxmin fair rates. Section 3.2 presents a framework for distributed implementation of the above algorithm. Section 4 presents another distributed algorithm, which operates under more general assumptions than the above implementation but has a slower convergence time in the worst case. Section 5 discusses various multirate mechanisms for allocation of maxmin fair rates to the receivers once the fair rates are computed. The concluding section, section 6 identifies some directions for future research.

## 2 Network Model

We consider an arbitrary topology network with  $N$  multicast sessions. A multicast session is identified by the triplet  $(n, v, U)$ ,  $n$  is a unique number assigned to the session,  $v$  is the source node of the session and  $U$  is the group of intended destination nodes ( $n$  has been incorporated to distinguish between different sessions with same source destination pair). We assume that the traffic from node  $v$  is transported across a predefined multicast tree to nodes in  $U$ . The tree can be established during connection establishment phase if the network is connection oriented or can be established by some well known multicast routing protocol like DVMRP[14], MOSPF[22], CBT[5], PIM[15] and MIP[26] in internet type network. The receivers may have minimum rate constraints. Also some sources may not be able to transmit at a rate higher than a certain threshold. Some receivers may have a low processing ability. In that case, it is useless to allocate higher rates to that session. So rate allocations can have peak rate constraints as well. These parameters are useful for ATM like scenarios, where session establishment is preceded by a negotiation stage and the network can be informed of these requirements during the negotiation stage. In a connectionless network, sessions can not have any such requirement as the network would never know of these, and would have to make rate allocation irrespective of any such requirement. Such a scenario can very well be accommodated in our model, by assuming minimum rate requirement as 0 and maximum rate to be  $\infty$  for each receiver.

We call every source destination pair of a session a virtual session. For example, if a session  $n$  has source  $v$  and destination set  $U$ , where  $U = \{u_1, \dots, u_t\}$ , then this session would correspond to  $t$  virtual sessions,  $(n, v, u_1), \dots, (n, v, u_t)$ . For example, the network shown in figure 1 has a single session,  $(1, v, U)$ , with  $U = \{u_1, \dots, u_4\}$ . This session corresponds to 4 virtual sessions,  $(1, v, u_1), (1, v, u_2), (1, v, u_3), (1, v, u_4)$ . Our objective would be to achieve a maxmin fair rate allocation for the virtual sessions. We look at rates attained by virtual sessions instead of those attained by the actual sessions because our requirement is that every receiver of every session should get a bandwidth commensurate with its fair share of the capacity of the path between the source and the receiver. So ensuring maxmin fairness of the session rate allocation and allocating the maxmin fair session rate to all the virtual sessions would cause intra-session unfairness. We assume that ev-

ery virtual session (source-destination pair) has a minimum and a maximum rate.

At this point, we would like to mention that by “rate” of a virtual session, we generally mean the average rate at which the corresponding receiver receives information. Because of the burstiness of traffic, it may not be a good idea to utilize the full capacity of the links. To prevent excessive delay and severe performance degradation, the network designer may like to have the average rates sum up to a certain fraction of the actual capacity of a link. In this context, the term “capacity of a link” which we use henceforth, stands for that fraction of the actual link capacity, which the network designer wants the sessions to use.

Informally speaking a rate allocation for the virtual session is feasible, if the rate for every virtual session is between the minimum and the maximum possible rates for the virtual session. Besides if session  $n$  corresponds to virtual sessions  $m_{n1}, \dots, m_{nt}$  in link  $l$ , then the maximum of the rates allocated to the virtual sessions  $m_{n1}, \dots, m_{nt}$  is the bandwidth consumed by session  $n$  in link  $l$ . Total bandwidth consumed by all sessions traversing through link  $l$  can not exceed the capacity of link  $l$ . More formally let there be  $M$  virtual sessions. A  $M$ -dimensional vector  $(r_1, \dots, r_M)$  is a feasible rate allocation if

1.  $\mu_i \leq r_i \leq p_i \forall i$ , where  $\mu_i$  and  $p_i$  are respectively the minimum and maximum rates of virtual session  $i$ ,  $p_i \geq \mu_i \geq 0$ ,
2. Let  $n(l)$  denote the set of sessions passing through link  $l$  and  $m(k, l)$  denote the set of virtual sessions of session  $k$  passing through link  $l$  and  $C_l$  denote the capacity of link  $l$ .  $\sum_{i \in n(l)} \max_{j \in m(i, l)} r_j \leq C_l$ .  $\lambda_{il}$  denotes the rate allocated to the session  $i$  on link  $l$  under rate allocation  $\vec{r}$ . It is actually the maximum of the rates allocated to the virtual sessions in  $m(i, l)$ , i.e.,  $\lambda_{il} = \max_{j \in m(i, l)} r_j$ . The capacity condition can also be stated as

$$\sum_{i \in n(l)} \lambda_{il} \leq C_l \quad (\text{capacity condition})$$

Figure 2 illustrates an example network with a few capacity and maximum and minimum rate constraints.

*Example 2.1:* The capacity of edge  $e_i$  is  $C_i$ .  $(C_1, \dots, C_6) = (7, 3, 6.5, 3, 4, 6)$  units.  $n(e_1) = n(e_3) = \{1, 2\}$ ,  $n(e_2) = n(e_4) = n(e_5) = \{1\}$ ,  $n(e_6) = \{2\}$ .  $m(1, e_1) = \{1, 2\}$ ,  $m(2, e_1) = \{3\}$ ,  $m(1, e_2) = \{1\}$ ,  $m(1, e_3) = \{2\}$ ,  $m(2, e_3) =$



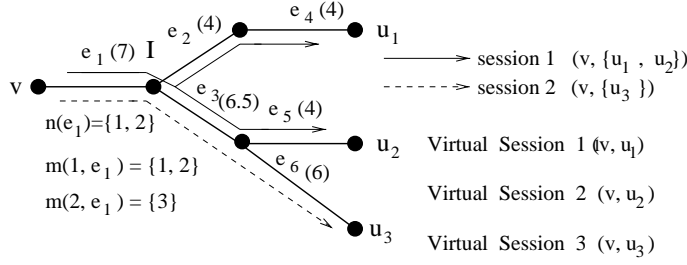


Figure 2: Session 1 includes virtual sessions, 1, 2. Session 2 includes virtual session 3. The figure shows some sample  $n(e_i)$  and  $m(i, e_j)$ s. The numbers in brackets,  $()$  denote the capacities of the respective links. The capacity constraint for link  $e_1$  is  $\max(r_1, r_2) + r_3 \leq 7$  and that for link  $e_3$  is  $r_2 + r_3 \leq 6.5$ . Virtual session 1 requires a minimum rate of 4 units and has a maximum rate of 5 units. Virtual session 2 has a minimum rate of 1 unit and maximum rate of  $\infty$  (no constraint on maximum rate). Virtual session 3 can have a minimum rate of 0 and a maximum rate of 5 units. Thus minimum and maximum rate constraints are  $4 \leq r_1 \leq 5$ ,  $1 \leq r_2 \leq \infty$  and  $0 \leq r_3 \leq 5$ .

$\{3\}$ ,  $m(1, e_4) = \{1\}$ ,  $m(1, e_5) = \{2\}$ ,  $m(2, e_6) = \{3\}$ . A rate vector  $(r_1, r_2, r_3)$  is feasible if

$$\left. \begin{array}{l} 2 \leq r_1 \leq 4 \\ 1 \leq r_2 \leq \infty \\ 0 \leq r_3 \leq 5 \end{array} \right\} \text{Minimum and Maximum rate constraints}$$

$$\left. \begin{array}{l} \max(r_1, r_2) + r_3 \leq 7 \quad (\text{Link } e_1) \\ r_1 \leq 3 \quad (\text{Link } e_2) \\ r_2 + r_3 \leq 6.5 \quad (\text{Link } e_3) \\ r_1 \leq 3 \quad (\text{Link } e_4) \\ r_2 \leq 4 \quad (\text{Link } e_5) \\ r_3 \leq 6 \quad (\text{Link } e_6) \end{array} \right\} \text{Capacity constraints} \quad (1)$$

Informally, a feasible rate vector is maxmin fair if it is not possible to maintain feasibility and increase the rate of a virtual session without decreasing that of any other virtual session which has equal or lower rate. More formally, a feasible rate allocation vector  $\vec{r}^1$  is maxmin fair if it satisfies the following property with respect to any other feasible rate allocation vector  $\vec{r}^2$ : if there exists  $i$  such that the  $i$ th component of  $\vec{r}^2$  is strictly greater

than that of  $\bar{r}^1$ , then there exists  $j$  such that the  $j$ th component of  $\bar{r}^1$ ,  $r_j^1$  is less than or equal to the  $i$ th component of  $\bar{r}^1$ ,  $r_i^1$  ( $r_j^1 \leq r_i^1$ ) and the  $j$ th component of  $\bar{r}^2$  ( $r_j^2$ ) is strictly less than the  $j$ th component of  $\bar{r}^1$  ( $r_j^2 < r_j^1$ ). The bandwidth allocations according to  $\bar{r}^2$  are less even than those according to  $\bar{r}^1$  in some sense.

*Example 2.2:* The maxmin fair rate vector in Example 2.1 is  $(4, 3.5, 3)$ . It is easy to check that this rate vector is feasible. It is not possible to decrease  $r_1$  below 4 because of the minimum rate constraint. This and the capacity constraint of link  $e_1$  forces  $r_3$  to be at most 3. Any increase in  $r_1$  or  $r_2$  will cause a decrease in  $r_3$  which is less than both. Thus  $(4, 3.5, 3)$  is the maxmin fair rate vector.

Henceforth we shall ignore the maximum rate constraints. This does not cause any loss in generality because maximum rate constraints can be incorporated by adding artificial links between receivers with maximum rate constraints and the rest of the network, with capacities of the artificial links equal to the maximum rates of the respective receivers. The size of the augmented network is comparable to that of the given network. So complexity of any algorithm for computation of the maxmin fair rates in a network should remain the same, if we use the augmented network instead.

Next we present a significant property of the maxmin fair rate vector. We first introduce the concept of bottleneck links. A link  $l$  is said to be *bottlenecked* with respect to a virtual session  $k$  traversing across it if the following conditions are met:

- Capacity of the link is fully utilized, i.e., the sum of the rates allocated to the sessions travelling across the link must be equal to the capacity of the link:  $\sum_{i \in n(l)} \lambda_{il} = C_l$
- The virtual session has the maximum rate amongst all virtual sessions of the same session travelling through the link, i.e.,  $r_k = \lambda_{\chi(k)l}$  where  $\chi(k)$  is the session corresponding to virtual session  $k$ .
- If any other virtual session  $j$  traversing through link  $l$ , has a rate higher than that of virtual session  $k$ , then rate of virtual session  $j$  is less than or equal to the minimum possible rate of some virtual session in  $m(\chi(j), l)$  (set of virtual sessions travelling across link  $l$  and belonging to the

same session as virtual session  $j$ ). In other words, if  $r_j > r_k$  then  $r_j \leq \mu_{\chi(j)l}$  where  $\mu_{\chi(j)l} = \max_{p \in m(\chi(j), l)} \mu_p$ . The maximum of the minimum rate requirements of the virtual sessions in  $m(i, l)$  is  $\mu_{il}$ .

*Example 2.3:* Refer to the network of Example 2.1. Consider the rate vector  $(4, 3.5, 3)$ . Link  $e_1$  is bottlenecked w.r.t. virtual session  $(v, u_1)$  and  $(v, u_3)$  and  $e_3$  is bottlenecked w.r.t. virtual session  $(v, u_2)$ . Link  $e_6$  is not bottlenecked w.r.t. any virtual session because its capacity is not fully utilized. Consider the rate vector  $(4, 4.5, 2)$ . This is also a feasible rate vector. Now no link is bottlenecked w.r.t. virtual session  $(v, u_3)$ .

The definition of a bottleneck link is similar to that in the unicast context. In the unicast context, a link is bottlenecked w.r.t. a session  $i$  if its capacity is fully utilized and if any other session  $j$  traversing the link has greater bandwidth, then the rate of  $j$  is equal to its minimum required rate. In multicast, the difference is that the rate of a different virtual session  $j$  can be more than that of  $s$  on the bottleneck link of  $s$ , not only because of its minimum rate requirement, but because of the minimum rate requirement of some other virtual session of the same session as  $j$ , traversing this bottleneck link. In absence of minimum rate requirements, the bottleneck condition becomes very simple for multicast networks as well. A link is bottlenecked w.r.t. a virtual session if its capacity is fully utilized and no other virtual session traversing the link has a greater rate.

**Lemma 1 (Bottleneck Lemma)** *A feasible rate vector is maxmin fair iff every virtual session has a bottleneck link.*

**Remark:** Bottleneck Lemma serves as a test for maxmin fairness of a feasible rate allocation vector. It indicates that if a rate vector is maxmin fair, then the rate of receiver  $s$  is at least  $C_l/|n(l)|$ , for some link  $l$  on its path, if there are no minimum rate requirements. In presence of minimum rate requirements, this lower bound becomes  $\frac{C_l - \sum_{i \in \tau(l)} \mu_{il}}{|n(l) \setminus \tau(l)|}$ , where  $\tau(l)$  is the set of sessions traversing link  $l$ , whose session rates on link  $l$  are greater than the rate of virtual session  $s$ . We will use this lemma in proving the correctness of an algorithm for computation of a maxmin fair rate allocation. We have proved this lemma in the appendix.

### 3 A synchronous distributed algorithm for computation of maxmin fair rates

We proceed to present a synchronous distributed algorithm for computing the maxmin fair virtual session rates shortly. It is synchronous in the sense that all sessions must start at the same time. We first give a generic description of the algorithm and then discuss its distributed implementation.

#### 3.1 Description of Algorithm

$n(l)$ ,  $m(i, l)$ ,  $\chi(s)$  and  $\mu_{il}$  are as defined in pages 7, 7, 9 and 10 respectively. We introduce some additional terminologies.

$L_s$  is the set of links traversed by virtual session  $s$

$r_s(k)$  is the bandwidth allocated to virtual session  $s$  at the end of the  $k$ th iteration.  $\vec{r}(k)$  denotes the rate vector at the end of the  $k$ th iteration, with components  $r_s(k)$ .

$\lambda_{il}(k)$  is the rate allocated to the session  $i$  on link  $l$  at the end of the  $k$ th iteration. It is actually the maximum of the rates allocated to the virtual sessions in  $m(i, l)$  at the end of the  $k$ th iteration.

A virtual session  $s$  is *saturated* under rate vector  $\vec{r}(k)$  if there exists a link  $l$  on its path such that the capacity of the link is fully utilized and  $s$  has the maximum rate amongst all virtual sessions of  $m(\chi(s), l)$ , i.e.,

$$\sum_{i \in n(l)} \lambda_{il}(k) = C_l \text{ and } r_s(k) = \lambda_{\chi(s)l}(k).$$

A session is *saturated* on a link  $l$  if all the virtual sessions of the session travelling through the link  $l$  are saturated.

$S(k)$  denotes the set of unsaturated virtual sessions at the end of the  $k$ th iteration.

$\Xi_l(k)$  denotes the set of unsaturated sessions passing through link  $l$  at the end of the  $k$ th iteration.

$F_l(k)$  denotes the total bandwidth consumed by the saturated sessions passing through link  $l$  at the end of the  $k$ th iteration.

$\eta_l(k)$  denotes the link control parameter of link  $l$  at the end of the  $k$ th iteration. Link control parameter is the iterate which would be used in computation of the maxmin fair rates. It is an estimate of the fair share of the bandwidth of the link which can be allocated to the unsaturated virtual sessions traversing the link. This bandwidth would have been allocated to the unsaturated virtual sessions traversing the link, if there were no bandwidth constraints on other links and also if the virtual sessions did not have any minimum rate constraints. As we shall see later, if a virtual session is saturated, then its rate is already determined.

$\eta_{il}(k)$  denotes the session link parameter of session  $i$  traversing through link  $l$ . It is the bandwidth assigned to session  $i$  if there were no bandwidth constraints for any of its virtual sessions on other links.

The following algorithm computes the maxmin fair rates for the virtual sessions.

1.  $k = 0$   $\eta_l(0) = 0$ ,  $F_l(0) = 0$ ,  $S(0) = \{1, \dots, M\}$ ,  $\Xi_l(0) = n(l) \forall$  link  $l$ ,  $r_s(0) = \mu_s \forall$  virtual session  $s$ .
2.  $k \rightarrow k + 1$
3. For every link  $l$  in the network compute the link control parameter. If  $\Xi_l(k-1) \neq \phi$ , then  $\eta_l(k)$  is the maximum possible  $\theta$ , which satisfies the equation,  $F_l(k-1) + \sum_{i \in \Xi_l(k-1)} \max(\theta, \mu_{il}) = C_l$  else  $\eta_l(k) = \eta_l(k-1)$ .  $\eta_{il}(k) = \max(\eta_l(k), \mu_{il})$ .\*
4. Compute  $r_s(k)$  for all virtual sessions  $s$ , where  $r_s(k) = \min_{l \in L_s} \eta_{\chi(s)l}(k)$ , if  $s \in S(k-1)$ , else  $r_s(k) = r_s(k-1)$ .
5. For every link  $l$  in the network compute the session rate in link  $l$ , for every session in  $n(l)$ ,  $\lambda_{il}(k) = \max_{s \in m(i,l)} r_s(k)$ .

---

\*This computation need be done for all unsaturated sessions traversing link  $l$  only.

6. Compute the set of virtual sessions unsaturated after the  $k$ th iteration,  $S(k) = S(k-1) \setminus \{s : \exists l \in L_s, \text{ s.t. } \sum_{i \in n(l)} \lambda_{il}(k) = C_l \text{ and } r_s(k) = \lambda_{\chi(s),l}(k)\}$ .
7. If  $S(k) = \phi$ , i.e., all virtual sessions are saturated, the algorithm terminates, else go to next step.
8. For every link  $l$ , compute the set of unsaturated sessions passing through the link  $l$  at the end of the  $k$ th iteration:  $\Xi_l(k) = \{n : n \in \{1, \dots, N\}, m(n, l) \cap S(k) \neq \phi\}$ .
9. For every link  $l$ , for which  $\Xi_l(k) \neq \phi$ , compute the bandwidth consumed by the saturated sessions passing through link  $l$ ,  $F_l(k) = \sum_{i \in n(l) \setminus \Xi_l(k)} \lambda_{il}(k)$ .
10. Go to step (2).

At every iteration  $k$ , the algorithm computes a “fair share” of the link bandwidth for every session,  $i$ , the session link parameter,  $\eta_{il}(k)$ . This bandwidth is offered to all virtual sessions of session  $i$  traversing the link. If a virtual session is constrained to have a rate less than its fair share because it is assigned a lower bandwidth on another link, then it can not use some of this bandwidth. If all virtual sessions of the same session release some bandwidth because of constraints on other links, then there is residual bandwidth. The residual bandwidth is split fairly among other sessions in the next iteration and the process continues. For deeper insight, ignore the minimum rate constraints for the time being. Initially all the link control parameters are assigned zero values. All sessions and virtual sessions are unsaturated. Next the algorithm computes the link control parameters as per step (3). The link control parameter for link  $l$  at 1st iteration, is the capacity of the link per session traversing the link. If a virtual session traversing link  $l$ , had no bandwidth constraint on other links, then it is assigned a rate equal to the link control parameter. On account of the bandwidth constraint in other links, the virtual session gets a rate equal to the minimum of the link control parameters on its path. All virtual sessions traversing through the link with the minimum link control parameter are saturated (this fact has been proved later). A session is saturated if all its virtual sessions are saturated. The bandwidth consumed by the saturated sessions, if any, are computed. This bandwidth is subtracted from the link capacity, and the link control parameters are recomputed at the beginning of every iteration. The link control

parameter in the next iteration is the residual capacity per unsaturated session traversing the link, if any, else it is the same as the link control parameter of the previous iteration. The virtual session bandwidths are assigned in the same manner and the process continues. Since at least one virtual session is saturated every iteration, the algorithm terminates in at most  $M$  iterations ( $M$  is the number of virtual sessions).

In presence of nonzero minimum rates, link control parameter of link  $l$  is computed as per step (3). In absence of bandwidth constraints on other links, a virtual session  $s$  traversing through link  $l$  is assigned a rate equal to its session link parameter on link  $l$ ,  $\eta_{\chi(s)l}(k)$ , where  $\chi(s)$  is the session of virtual session  $s$ . Session link control parameter is the maximum of the link control parameter and the minimum allowable rates ( $\mu_{s}$ ) of the virtual sessions of the session traversing through the link. On account of the bandwidth constraint in other links, the virtual session gets a rate equal to the minimum of the above quantity, the minimum taken over all links in the path of the virtual session. Again, at least one virtual session is saturated every iteration (proved later), and the process continues till all virtual sessions are saturated.

We illustrate the operation of the algorithm with an example.

*Example 3.1:* Consider the network of Example 2.1. The minimum rate for virtual session  $(v, u_1)$  is now 4. The capacities of edges  $e_2, e_4$  are now 4. The maximum rate constraints do not exist. The rest of the constraints remain the same. Virtual sessions  $(v, u_1), (v, u_2)$  and  $(v, u_3)$  are named virtual sessions 1, 2, 3 respectively. Virtual sessions 1, 2 belong to session 1 and virtual session 3 belongs to session 2.  $L_1 = \{e_1, e_2, e_4\}$ ,  $L_2 = \{e_1, e_3, e_5\}$ ,  $L_3 = \{e_1, e_3, e_6\}$ . Link control parameters are as follows.  $\eta_{e_1}(1) = 3$ ,  $\eta_{e_2}(1) = 4$ ,  $\eta_{e_3}(1) = 3.25$ ,  $\eta_{e_4}(1) = 4$ ,  $\eta_{e_5}(1) = 4$ ,  $\eta_{e_6}(1) = 6$ . Now, the session link control parameters are as follows.  $\eta_{1e_1}(1) = 4$ ,  $\eta_{2e_1}(1) = 3$ ,  $\eta_{1e_2}(1) = 4$ ,  $\eta_{1e_3}(1) = 3.25$ ,  $\eta_{2e_3}(1) = 3.25$ ,  $\eta_{1e_4}(1) = 4$ ,  $\eta_{1e_5}(1) = 4$ ,  $\eta_{2e_6}(1) = 6$ . Computing the  $r_s(1)$ s as per step 4, we have  $r_1(1) = 4$ ,  $r_2(1) = 3.25$ ,  $r_3(1) = 3$ . Observe that virtual sessions 1 and 3 are saturated, while virtual session 2 is not.  $S(1) = \{2\}$ . Thus session 2 is saturated on all links. Session 1 is unsaturated on only those links which are on the path of virtual session 2.  $\Xi_{e_1}(1) = \Xi_{e_3}(1) = \Xi_{e_5}(1) = \{1\}$ ,  $\Xi_l(1) = \phi$ , if  $l \notin \{e_1, e_3, e_5\}$ .  $F_{e_1}(1) = F_{e_3}(1) = 3$ , and  $F_{e_5}(1) = 0$ . Computations for the next iteration are as follows.  $\eta_{e_1}(2) = 4$ ,  $\eta_{e_3}(2) = 3.5$ ,  $\eta_{e_5}(2) = 4$ .  $\eta_l(2) = \eta_l(1)$

for the rest of the links.  $\eta_{1e_1}(2) = 4$ ,  $\eta_{1e_3}(2) = 3.5$ ,  $\eta_{1e_5}(2) = 4$ . Thus  $r_2(2) = 3.5$ .  $r_s(2) = r_s(1)$ ,  $s \in \{1, 3\}$ . Now virtual session 2 is also saturated. So  $S(2) = \phi$  and the algorithm terminates. The rates obtained upon termination are  $(4, 3.5, 3)$ .

The rates allocated to the virtual sessions, upon termination of the algorithm are the maxmin fair rates. This follows from the following theorems.

**Theorem 1** *If the algorithm terminates in  $k$  iterations, then  $\vec{r}(k)$  is the max-min fair rate vector.*

The formal proof is presented the appendix. The intuition behind the result is as follows: max-min fair sharing implies that if there are  $k$  sessions sharing a link, each session should get a “fair share” of the link bandwidth. If a session is constrained to have a rate less than its fair share because it is assigned a lower bandwidth on another link, then the residual bandwidth is split fairly among other sessions. This is exactly what the algorithm does. The last theorem ensures that the algorithm terminates in finite number of iterations.

**Theorem 2** *The algorithm terminates in at most  $M$  iterations, where  $M$  is the number of virtual sessions.*

We prove this theorem in appendix. The proof considers the link which attains the minimum link control parameter in the  $k$ th iteration amongst all those links which carry at least one unsaturated session. We show that the capacity of the link is fully utilized, and at least one unsaturated virtual session traversing the link has rate equal to its session rate on the link. This virtual session saturates in the  $k$ th iteration. Thus all virtual sessions saturate by  $M$  iterations.

Some well known algorithms for computation of maxmin fair rates in the unicast scenario, e.g., [3], [6], are special cases of this algorithm. The simplest version of this algorithm for unicast networks is in [6]. Later [3] and [18] generalized this algorithm to take care of minimum rate constraints in ATM ABR unicast networks. We have considered a multicast network with minimum and maximum rate constraints. The introduction of multicast complicates the problem significantly. For example, the basic constraints for feasibility are different. We need to think in terms of virtual sessions instead



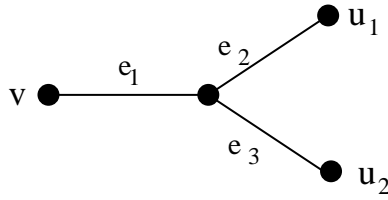


Figure 3:

of sessions. The condition for a link to be bottlenecked w.r.t. a virtual session is different from that of a link to be bottlenecked w.r.t. a session in the unicast scenario. Also a virtual session can have a rate higher than the link control parameter of a link it traverses, not only if its minimum bandwidth requirement is higher, but also if the minimum bandwidth requirement of some other virtual session of the same session traversing through the same link is higher. The generalization and the proof need to take care of these intricate details. For example, the algorithm for computation of maxmin fair rates would terminate in at most  $|\mathcal{L}|$  iterations, for unicast networks, where  $\mathcal{L}$  is the set of links. The same does not hold for multicast networks. Consider the following example.

*Example 3.2:* Consider the network shown in Figure 3. There are 4 multicast sessions, all of them having node  $v$  as source and nodes  $u_1$  and  $u_2$  as destinations. Thus there are 8 virtual sessions,  $1, \dots, 8$ . Virtual session  $j$  corresponds to session  $\lceil \frac{j}{2} \rceil$ . Virtual sessions 1, 3, 5, 7 have  $u_2$  as destination and the rest have  $u_1$  as destination. Node  $v$  is the source of all virtual sessions. The minimum rate requirements of virtual sessions 1, 3, 5, 7 are 1, 2, 2.25, 2.5 units. The rest of the virtual sessions have 0 as the minimum rate requirement.  $C_{e_1} = 7.75$ ,  $C_{e_2} = 7.6$  and  $C_{e_3} = 16$  units. Without going into the details of the computation,  $\eta_{e_1}(1) = 1$ ,  $\eta_{e_2}(1) = 1.9$ ,  $\eta_{e_3}(1) = 4$ ,  $\vec{r}(1) = (1, 1, 2, 1.9, 2.25, 1.9, 2.5, 1.9)$ . Virtual sessions (1, 2, 3, 5, 7) are saturated at the end of the first iteration.  $\eta_{e_1}(2) = 2$ ,  $\eta_{e_2}(2) = 2.2$ ,  $\vec{r}(2) = (1, 1, 2, 2, 2.25, 2.2, 2.5, 2.2)$ . Virtual sessions 6 and 8 remain unsaturated.  $\eta_{e_1}(3) = 2.25$ ,  $\eta_{e_2}(3) = 2.3$ ,  $\vec{r}(3) = (1, 1, 2, 2, 2.25, 2.25, 2.5, 2.3)$ . Virtual session 8 remains unsaturated.  $\eta_{e_1}(4) = 2.5$ ,  $\eta_{e_2}(3) = 2.35$ ,  $\vec{r}(4) = (1, 1, 2, 2, 2.25, 2.25, 2.5, 2.35)$ . All virtual sessions are saturated at the end of the 4th iteration. The algorithm terminates in 4 iterations, though there are

3 links.

The algorithm terminates in at most  $|\mathcal{L}|$  iterations for multicast networks, if a specific condition is met, i.e., if all virtual sessions of the same session sharing a link at some point, have the same minimum rate requirement ( $\mu_i = \mu_j$  if  $\chi(i) = \chi(j)$  and  $L_i \cap L_j \neq \phi$ ) (Lemma 2). Note that this condition on minimum rates always holds in unicast networks because every session has only one virtual session.

**Lemma 2** *The algorithm terminates in at most  $\min(|\mathcal{L}|, M)$  iterations, if  $\mu_i = \mu_j$  for all  $i, j$  such that  $\chi(i) = \chi(j)$  and  $L_i \cap L_j \neq \phi$ .*

We prove this lemma formally in appendix.

Every step of this algorithm has a complexity of  $O(|\mathcal{L}|M)$ . The algorithm must terminate in  $M$  iterations (Theorem 2). Thus the overall complexity of this algorithm is  $O(|\mathcal{L}|M^2)$ .

We describe the distributed implementation of this algorithm in the next subsection.

### 3.2 Distributed Implementation of the Synchronous Algorithm

We describe a framework for synchronous distributed implementation of the algorithm presented in the previous subsection. We will exploit the fact that the computation of the session link control parameters of any link needs only information about the saturation status of sessions traversing the link and the previous iteration rates of the unsaturated sessions traversing the link. A virtual session (receiver) can determine its rate if it knows its session link control parameters of the links on its path. To know whether it is saturated or unsaturated, it needs to know only the session link rates on its path.

Every node maintains an information field for every outgoing link. The information field has an entry for the link control parameter and a separate record for every session traversing the link. The record for session  $i$  traversing link  $l$  maintains the session rate, the minimum required rate for the session, session saturation map and a rate bitmap for the session. The session saturation map has one entry for every virtual session of  $m(i, l)$ . The entry for virtual session  $s \in m(i, l)$  indicates whether it is saturated or unsaturated or

its status is not known. The rate bitmap has one bit for every virtual session of  $m(i, l)$ . The entry for virtual session  $s \in m(i, l)$  indicates whether its rate is known or not known.

At first every entry in the saturation maps indicates unsaturated, and every entry in the rate bitmaps indicates that rate is known. All session rates are initialized to the minimum rates. Source rates are initialized to the minimum allowable rates (maximum of the minimum rates required by all receivers of the session). Consider link  $l$ . Link control parameter is computed as per step (3) at the origin of link  $l$ . Next the session link control parameters are computed for every unsaturated session traversing the link. Next, the entries for unsaturated virtual sessions in the saturation map and the rate bitmap for the sessions are changed to “dont know” values. The link now waits for “rate packets” from the virtual sessions with “dont know” rate bits. The unsaturated virtual sessions send backward rate packets towards the source with very high rate values. If a backward rate packet from a receiver reaches a node while the receiver entry in the rate bitmap shows that rate is known, then this means that the rate packet needs to know the rate computed in the next iteration, while the node has not yet performed the next iteration computation. In this case, the node holds the rate packet, till the corresponding entry in the rate bitmap indicates that rate is not known and then it decreases the rate value in the packet to the session link control parameter,  $\eta_{\chi(s)l}$ , if the rate packet reaches it through link  $l$  and if the rate value of the packet is greater than the session link control parameter. Subsequently, the node forwards the rate packet towards the source. After the rate packet reaches the source, the source updates its rate to the maximum of the current rate and the rate value in the rate packet and subsequently generates a forward rate packet towards the receiver, with the same rate value. The forward rate packet contains the minimum session link control parameter on the path of the virtual session and this value is the current iteration rate of the virtual session. When a forward rate packet reaches the node, it updates the rate bit to indicate that rate is known, and updates the session rate to maximum of the rate in the rate packet and the existing session rate. When the forward rate packet reaches the receiver, the receiver knows its current iteration rate and generates a probe message to query its saturation status. The probe message contains the current iteration rate of the virtual session. The probe message travels towards the source. When the probe message reaches a node through link  $l$ , the

node holds the probe message till all entries in the rate bitmap for link  $l$  indicates that the rates are known and then if the node determines that link  $l$  satisfies the conditions for saturation, it updates the corresponding entry in the saturation map in its information field and generates forward and backward saturation messages. The forward message travels towards the receiver and the backward message travels towards the source. If the node determines that the link does not satisfy the conditions for saturation, it forwards the probe message towards the source. If the probe message reaches the source, then the source sends a forward unsaturation message towards the receiver. Whenever a node receives a forward or backward, saturation or unsaturation message, it updates its saturation bitmap accordingly. After a virtual session receives an unsaturation message, it sends a new backward rate packet, because unsaturation message means that all nodes on its path know the saturation status and rates of all virtual sessions and sessions respectively and is thus ready for another iteration computation. Unsaturation message also means that the virtual session has not received its final rate. If a virtual session receives a saturation message, then it knows its final rate and it does not generate any further rate packet. When all entries in the saturation maps are updated to indicate saturation or unsaturation for all sessions traversing a link, and at least one entry is unsaturated, then the origin node of the link computes the link control parameter for another iteration and the process repeats. If all entries are saturated, then the node does not make any further computation.

The distributed implementation terminates in  $2DM$  units of time where  $D$  is the maximum round trip time from a source to a receiver in the network. This is because after a link control parameter is computed, rates of all virtual sessions traversing the node are known at the node in  $D$  units of time. A probe is sent at or before this time and thus the saturation or unsaturation message reaches the receivers and hence the nodes on the way at most  $D$  units after this time. Thus a new iteration can start in  $2D$  units of time after the first. As Theorem 2 indicates, any node needs to perform at most  $M$  iterations.

We have described only a framework for distributed computation. This description gives the necessary intuition. Various optimizations can be introduced to make the computation more efficient. For example, it is not actually necessary to maintain a separate entry for every virtual session of a session in its saturation and rate maps. One entry per session per link for

rate and saturation status of the session is sufficient. Besides, the rate and probe packets of different virtual sessions of the same session can be merged at branching points, so that there is only one rate (probe) packet of a session in a link. This prevents control information implosion at the source. We describe an implementation with all the above modifications below.

Now every node maintains an information record for each of its outgoing links  $l$ . The record maintains the following entries for every session  $i$  traversing  $l$  :

Minimum session link rate:  $\mu_{il}$

Rate bit

Rate value

Saturation entry

Note that the entry does not maintain separate information about different session  $i$  virtual sessions traversing link  $l$ . If the rate bit is set, then the current iteration session link rate is known at the node. The rate value is the current iteration session link rate. The saturation entry has three possible values to indicate whether the session is saturated or unsaturated in the link or its status is not known. Initially, all rate bits indicate that rates are known. Also all saturation entries indicate that sessions are not saturated. All receivers are unsaturated. Consider link  $l$ . Link control parameter is computed as before. Subsequently, all rate bits are reset and all saturation entries which indicate “unsaturated” are updated to reflect “dont know” value. The origin of the link waits for backward rate packets of unsaturated sessions. Unsaturated receivers transmit backward rate packets towards the respective source, with very high rate values. When a node receives a session  $i$  backward rate packet along link  $l$ , it holds it till session  $i$  rate bit on link  $l$  is reset. After, the session  $i$  rate bit is reset, it updates the session  $i$  rate value in link  $l$  to the minimum of the session link control parameter and the rate value in the rate packet. The node waits for session  $i$  backward rate packets on other links originating from the node on session  $i$  path. Once all of them arrive, the node generates a backward rate packet with rate value equal to the maximum of the session  $i$  rate values stored in the node. The backward rate packet moves toward the source. When the source receives a backward

rate packet, it sets its transmission rate value equal to the rate value in the packet and sends a forward rate packet downstream with rate value equal to that of the backward rate packet. When a node receives a session  $i$  forward rate packet, it updates the session  $i$  rate values in each of its outgoing links on the path of session  $i$ . The new session  $i$  rate value in a link is the minimum of the old value and the rate value in the forward rate packet. Subsequently, session  $i$  rate bits are set on all links on session  $i$  path. Also, the node sends a forward rate packet on each of its outgoing links on session  $i$  path. The rate value of the forward rate packet is equal to the new session  $i$  rate value in the link. When a receiver receives a forward rate packet, it records its rate and sends a probe packet towards the source to query its saturation status. The probe packet contains the following entries:

status

test bit

The status entry has three possible values: “saturation,” “unsaturation” or “dont know.” If the status entry indicates saturation, then all downstream receivers are saturated. If the status entry indicates “unsaturation”, then at least one downstream receiver is unsaturated and hence the session is unsaturated on the link. If the status entry indicates “dont know,” then the saturation status of some downstream receiver is not known, and all downstream receivers whose saturation status are known, are saturated. If the test bit is set, then the node which receives the probe packet, must test saturation condition. If the test bit is reset, then no such test need be conducted as saturation status of all downstream receivers are known. If the status indicates unsaturated, then one downstream receiver is unsaturated, and hence the session is unsaturated on the link, but the saturation status of all downstream receivers may not be known yet. The test bit becomes useful in this scenario. It indicates whether saturation status of all downstream receivers are known or not.

The receivers send probe packets with test bit set and status indicating “dont know.” When a node  $n$  receives a session  $i$  probe packet through link  $l$ , it holds it till rate bits of all sessions traversing the link are set. Once all other session rates are known on the link, if test bit of the probe packet is set, the node determines whether the link capacity is fully utilized.

1. If the link capacity is fully utilized, then it resets the test bit and additionally if the status indicates “dont know” then it updates the status to saturation.
2. If the link capacity is not fully utilized but the session link rate is strictly less than the maximum of the session link rates for all links originating from the node and on session  $i$  path, it resets the test bit and additionally if the status indicates “dont know” it updates the status to “unsaturation”.

If the status indicates “saturation” or “unsaturation,” then the status is not altered in either of the two cases. If the test bit is reset, then the node does not check for saturation and test bit and status are not altered. The session  $i$  saturation entry in the link is set equal to the status of the probe packet. Next the node waits for the arrival and similar operation on session  $i$  probe packets on other links originating from  $n$ , on session  $i$  path and whose saturation entry indicated “dont know” just after the last link control parameter computation. Next, the node generates a session  $i$  probe packet and sends it towards the source of session  $i$ . If the test bit of at least one of the incoming probe packets remains set after modification, then test bit of the new probe packet is set. Otherwise, the test bit is reset. If

1. status entries of at least one incoming session  $i$  probe packet indicates unsaturation after modification or
2. status entry of at least one incoming session  $i$  probe packet indicates “dont know” and session  $i$  rate in the respective link is strictly less than the maximum of session  $i$  link rates for links on session  $i$  path and originating from the node,

then the status entry of the new probe packet is set “unsaturated,” else if at least one of the session  $i$  incoming probe packets indicate “dont know” after modification, then the status entry of the new probe packet is set “dont know”. Otherwise, the status entry is set “saturated.”

When the source receives the probe packet it generates a “saturation” or an “unsaturation” message. A saturation message is generated if the test bit is reset. Otherwise, an unsaturation message is generated. The message is sent downstream. When a session  $i$  saturation or unsaturation message arrives at a node, the node updates the saturation status of session  $i$  on the links

on its path, showing “dont know” values. If a saturation message is received, then the saturation status is modified to indicate saturation on all such links. If an unsaturation message is received, then the saturation status is modified to indicate “unsaturated” on all such links. If the saturation status does not indicate “dont know” value on a link, then it is not altered. The node generates a similar message on each of its outgoing links which are on the path of session  $i$ . The message is a saturation(unsaturation) message if the saturation status of session  $i$  on the link indicates saturation(unsaturation).

If a receiver receives an unsaturation message, it sends a new backward rate packet, with a very high rate value. If it receives a saturation message, then it does not send any further rate packet.

If a session is saturated in a link in an iteration, its saturation entry indicates saturation and if it is not saturated then its saturation entry indicates so. Also session link rates equal the rate values in the session fields in the links at the end of every iteration. Thus, this implementation exactly emulates the algorithm of Section 3.1. Note that here the nodes maintain information about session link rates and session saturation status only. It does not require explicit information about all virtual sessions which traverse the node. Besides, there is only one forward rate packet, backward rate packet, probe packet and saturation (unsaturation) message for a session in a link in one iteration, independent of the number of receivers downstream. This merger of control information prevents control information implosion. One can also think of other modifications which should reduce the computation time in most cases. For example, a saturation (unsaturation) message can be generated immediately after the probe packet status reflects saturation (unsaturation), instead of waiting for the probe packet to reach the source. This will not affect the worst case computation time, but should speed up the computations in most cases. The worst case computation time is  $2DM$  units like the previous implementation.

An advantage of this algorithm is that the intermediate rates are always feasible (Lemma 6). So during the computation period, sources can still transmit at intermediate rates, without any huge queue buildup. The disadvantage of this distributed computation is that it requires all sessions to start at the same time. Hence we call it a synchronous algorithm. That may not be the case. Besides, multicast group membership is dynamic in internet. Receivers join and leave during the lifetime of a session. We present another distributed algorithm for computation of maxmin fair rates which is



more suitable for a dynamic scenario. The essential principle behind the two algorithms is the same, but there are certain philosophical differences. We describe it in the next section.

## 4 An asynchronous distributed algorithm for computation of maxmin fair rates

We briefly describe the asynchronous algorithm below. For details, refer to the pseudocode in the next subsection.

### 4.1 Generic Description

Every source sends a forward rate packet with a rate value ( $r_p$ ), which is initialized to a very high value initially. The forward rate packet contains a state bit ( $u_p$ ). The state bit indicates whether the source should increase its rate or not. The source sets the state bit to 0. Every node maintains a link record for every outgoing link. The link record maintains an estimate of the link control parameter ( $\psi_l$ ), and a session field for every session traversing the link  $l$ . The session field of session  $i$  on link  $l$  on its path contains the forward and backward session link rates ( $f_{il}$  and  $b_{il}$  respectively), and a session link saturation bit ( $w_{il}$ ), for the session. Whenever a node receives a forward rate packet of a session, it performs the following sequence of actions for each of its outgoing links on the path of the session (for example, if a session traverses links  $l_1$  and  $l_2$  originating from node  $n$ , then it performs this sequence of actions for both  $l_1$  and  $l_2$ ):

1. Generates a forward rate packet. The rate value of the forward rate packet on the link is equal to the minimum of the rate value of the incoming forward rate packet and backward session link rate on the link. State bit of the new forward rate packet is initially reset.
2. If the rate value in the newly generated rate packet is less than the estimated link control parameter, then the session link saturation bit is set. If the rate value in the newly generated rate packet is greater than or equal to the estimated link control parameter, then the state bit of the newly generated rate packet is set and the rate value of the new rate packet is set equal to the estimated link control parameter.

3. Copies the rate value of the new forward rate packet to the forward session link rate entry and transmits the forward rate packet in the respective outgoing link.
4. Computes a new estimate of the link control parameter. This computation is done as follows when there are no minimum rate requirements. Refer to the pseudocode for the estimation procedure in presence of minimum rate requirements. Available capacity of a link is the difference between the link capacity and the sum of the forward session link rates of saturated sessions (sessions with session link saturation bit set to 1).
  - (a) If no session traverses a link, then link control parameter is equal to the link capacity.
  - (b) If all sessions traversing the link have their session link saturation bits set, then link control parameter is equal to the sum of the available capacity of the link and the maximum session link rate, the maximum taken over all sessions traversing the link.
  - (c) Otherwise, the link control parameter is the available capacity per unsaturated session traversing the link ( a session is unsaturated on a link if its session link saturation bit is reset).

After the link control parameter is computed, the session link saturation bits of sessions with forward session link rate greater than or equal to the link control parameter estimate are reset. Computation of link control parameter is repeated with the new set of saturated sessions, if the set of saturated sessions change. We do not need to repeat this computation more than twice.

Once the forward rate packet reaches the receiver, it records the rate value in the packet as its current iteration rate and generates a feedback rate packet. The rate value of the feedback rate packet is equal to the maximum of the rate value of the forward rate packet and the minimum rate of the receiver. The state bit of the feedback rate packet is equal to that of the forward rate packet. The minimum rate entry of the feedback rate packet contains the receiver minimum rate. When a feedback rate packet reaches a node through link  $l$ , the node updates the backward session link rate field for the session

on the link  $l$  respectively. The backward session link rate field is set equal to the rate value in the feedback rate packet, if the state bit of the backward rate packet is set, otherwise the backward session link rate is set to  $\infty$ . The node also updates its estimated minimum session link rate to the minimum rate value of the feedback rate packet. Then the node waits for the arrival of feedback rate packets of the same session on other links originating from the node. Once all of them arrive, the node generates a feedback rate packet towards the source of the session. The rate value of this feedback rate packet is equal to the maximum of the rate values of the incoming feedback rate packets, if all the feedback rate packets have the state bit set, otherwise, this rate value is set equal to the maximum of the rate value of the forward rate packet which reached the node from the source of the session and the minimum session link rates on the links which originate from the node and are on the path of the session. If the state bit of the same forward rate packet were set or all of the state bits of the incoming feedback rate packets are set, then the state bit of the new feedback rate packet is 1, otherwise it is 0. When the feedback rate packet reaches the source, the source generates another forward rate packet. If the state bit of the feedback rate packet is 0, the new forward rate packet has a very high value of rate (ideally  $\infty$ ). If the state bit is 1, the rate value in the new forward rate packet is equal to the rate value of the feedback rate packet.

Sessions can enter and exit, receivers can join existing sessions, or leave continuing sessions and link capacities can change any time during the execution of the algorithm. When new sessions join, or links are added to the path of an existing session on account of new receiver joins, the forward and the backward session link rates are initialized to very high values, typically  $\infty$ . A session is initially considered unsaturated on all newly added links on its path and the link control parameter of the links on its path are computed. When a session exits, or links are removed from its path on account of receiver leave events, link control parameters on its path are also computed.

**Theorem 3** *If the system stabilizes<sup>†</sup> at time  $t = t_0$ , then for all  $t \geq t_0 + 2.5D + 5DM$ , the receiver rates are equal to the maxmin fair rates. The forward session link rates are equal to the maxmin fair session link rates for all  $t \geq t_0 + 2.5D + 5DM$ .*

---

<sup>†</sup>System stabilizes means link capacities and minimum rate requirements do not change or sessions and their group members do not enter and exit any further.

The intuition behind the convergence result is that the state bit in the rate packet and the session link state bits indicate whether session link rates can be increased or not. If the estimated rate (rate value in the rate packet), is less than the fair share of the link capacities (link control parameters) on its path, then the rate packet returns with state bit reset and the rate estimate is increased next time. If the estimated rate is greater than or equal to the fair share of some link on its path, then the estimated rate is reduced to this fair share and the state bit is set, so that the next rate packet does not contain a greater estimate of the rate. The session link rates recorded at the nodes (forward session link rates) are set to the most recent value of the estimate (rate value in the most recent forward rate packet). Initially, the estimate is set to a very high value (typically,  $\infty$ ). Subsequently, increase and decrease triggered by the state bits bring the convergence of the session link rates to the fair shares and consequently the receiver rates converge to the respective fair shares. In absence of minimum rate requirements, the convergence is attained before  $t_0 + 5DM$ . Theorem 3 is formally proved in the appendix.

The advantage of this algorithm is that, unlike the synchronous algorithm, it need not restart every time there is a small change in the system. Thus, in practice, the algorithm may be faster than the synchronous one in a dynamic scenario, e.g., if multicast session membership changes frequently. The disadvantage of this algorithm is that its worst case convergence time is worse than that of the synchronous one. Besides, it needs to send control packets and estimate link control parameters all along the operation of the system. The synchronous algorithm need not send any control packets nor make any computation after the virtual sessions are saturated, till any further change in the system. So the synchronous algorithm is a better choice if changes are infrequent. Another disadvantage of the asynchronous algorithm is that the intermediate virtual session rates may not be feasible and this may buildup huge queues in transience period. This does not happen for the synchronous algorithm. Depending on the requirements of the system, any one of the above algorithms may be chosen.

A distributed asynchronous algorithm for computation of maxmin fair rates in unicast network, presented in [9], is a special case of ours. The algorithm in [9] does not take care of minimum rate constraints. However, [3] proposes a distributed asynchronous algorithm for computation of maxmin fair rates in unicast networks with minimum rate requirements. But, the algorithm is not guaranteed to converge in finite time. Another algorithm for

the same objective has been proposed in [18]. The convergence proof for this algorithm is in error. We point out the errors in the appendix. Our algorithm computes maxmin fair rates in multirate multicast networks with minimum rate requirements and is guaranteed to converge in finite time after the system stabilizes. This is the first such algorithm in multirate multicast networks with or without minimum rate constraints. If the network is unicast, then it is not necessary to keep track of the backward session link rates at the nodes at all. The algorithm in [9] maintains only the forward session link rates and saturation bits for sessions at the nodes. However, we need to keep track of this additional information for multicast networks (at least at nodes where fanout is greater than 1) to account for the subtlety that a session rate in a link may be greater than rates of some session receivers downstream. Our algorithm differs from [9] in another aspect as well. We set a session link saturation bit, only if forward session link rate is strictly less than the link control parameter estimate. However, [9] “marks” a session on a link (“marking” a session on a link is similar to setting the session link saturation bit) if the session rate is less than or equal to the link control parameter (advertized rate in their terminology). This difference is due to minimum rate requirements. The algorithm oscillates if sessions are marked when session rate is equal to the link control parameter in presence of minimum rate requirements[18]. We follow the remedy suggested by [18], i.e., we “mark” a session on a link only if the session link rate is strictly less than the link control parameter. However, this complicates the approach. The proof in [18] is not able to handle the complications in the unicast case. In view of this, the unicast version of our algorithm (assuming  $b_{il} = \infty$ , all along) is different from [18].

## 4.2 Pseudocode for asynchronous distributed algorithm

First we introduce some terminologies.

$\rho_i$  is the rate of transmission of session  $i$ .

$\sigma_s$  is the rate of reception of receiver  $s$ .

$r_p$	Rate Value of Rate Packet	} Rate Packet
$u_p$	State Bit of Rate Packet	
$\mu_p$	Minimum Rate	

A forward rate packet has only the first two entries. A feedback rate packet has all three of them.

$v_i^n$	Forward Session bit for session $i$	}	(a)
$g_i^n$	Incoming session rate for session $i$		
$x_i^n$	Feedback Session bit for session $i$		
$y_i^n$	Feedback outgoing session rate for session $i$		
$\tau_i^n$	Set of links originating from node $n$ on the path of session $i$		
$f_{il}$	Forward Session link rate	}	(b)
$b_{il}$	Backward Session link rate		
$w_{il}$	Session link saturation bit		
$h_{il}$	Session link Hold bit		
$\hat{\mu}_{il}$	Minimum session link rate estimate		

(a) gives the information field for session  $i$  on node  $n$ , if session  $i$  traverses node  $n$ . (b) gives the information field for session  $i$  on outgoing link  $l$  of node  $n$ , if session  $i$  traverses link  $l$ .

$\psi_l$	Estimate of link control parameter	}	Link $l$ record
$\kappa_l$	Set of sessions traversing link $l$		

$\vec{f}^l$  is the forward session link rate vector, with components  $f_{ils}$ .

$\vec{w}^l$  is the session link saturation bits vector, with components  $w_{ils}$ .

$\vec{\mu}^l$  is the minimum session link rates vector, with components  $\hat{\mu}_{ils}$ .

|( & ) indicates logical or (and) operation.

The pseudocode follows:

At source of session  $i$ :

1. If this is the first rate packet,  $\rho_i = \infty$ .
2. Otherwise, when a feedback rate packet( $r_p, u_p, \mu_p$ ) is received,
  - (a) If  $u_p = 0$ ,  $\rho_i = \infty$ .
  - (b) If  $u_p = 1$ ,  $\rho_i = r_p$ .

**generate-forward-rate-packet**( $\rho_i, 0$ ) and send it downstream.

At node  $n$  :

When session  $i$  forward rate packet( $r_p, u_p$ ) is received:

1.  $g_i^n = r_p$ ,  $v_i^n = u_p$ ,  $x_i^n = 1$ ,  $y_i^n = 0$ , and  $\forall l \in \tau_i^n$ , perform the following actions:
2.  $h_{il} = 1$ ,  $z_{il1} = \min(b_{il}, r_p)$ ,  $z_{il2} = 0$ ,
3. If  $z_{il1} < \psi_l$ ,  $w_{il} = 1$  and if  $z_{il1} \geq \psi_l$ ,  $z_{il1} = \psi_{il}$  and  $z_{il2} = 1$ . Here,  $\psi_{il} = \max(\psi_l, \hat{\mu}_{il})$ .
4.  $f_{il} = z_{il1}$ ,
5. **generate-forward-rate-packet**( $z_{il1}, z_{il2}$ ) and transmit it on link  $l$ , and
6. **estimate-link-control-parameter**( $f^l, \vec{w}^l, \vec{\mu}_l, C_l$ )

$z_{il1}$  and  $z_{il2}$  are local variables.

When session  $i$  feedback rate packet( $r_p, u_p, \mu_p$ ) reaches node  $i$  via link  $l$ ,

1.  $\hat{\mu}_{il} = \mu_p$ ,
2. If  $u_p = 1$ ,  $b_{il} = r_p$ , else  $b_{il} = \infty$ ,
3.  $x_i^n = x_i^n \& u_p$ ,
4. If  $u_p = 0$ ,  $y_i^n = g_i^n$  else  $y_i^n = \max(y_i^n, r_p)$ ,
5.  $h_{il} = 0$ ,
6. If  $\sum_{l \in \tau_i^n} h_{il} = 0$ , **generate-feedback-rate-packet**( $\max(y_i^n, \max_{l \in \tau_i^n} \hat{\mu}_{il}), v_i^n | x_i^n, \max_{l \in \tau_i^n} \hat{\mu}_{il}$ ) and send this feedback rate packet towards the source of session  $i$ .

At a receiver  $s$  of session  $i$  :

When a forward rate packet( $r_p, u_p$ ) is received,  $\sigma_s = \max(r_p, \mu_s)$  and **generate-feedback-rate-packet**( $\sigma_s, u_p, \mu_s$ ) and send it towards the source of session  $i$ .

Let  $n$  be the origin node of link  $l$  and let link  $l$  be added to the path of a session  $i$ . Session initiation can cause addition of links on its path. A link

may also be added to the path of an existing session on account of some new receiver joins.

1.  $\hat{\mu}_{il} = 0$ ,  $w_{il} = 0$ ,  $\kappa_l = \kappa_l \cup \{i\}$ ,
2. If session  $i$  was not traversing node  $n$ , prior to addition of link  $l$ ,  $\tau_i^n = \{l\}$ , else  $\tau_i^n = \tau_i^n \cup \{l\}$
3. **estimate-link-control-parameter** $(\vec{f}^l, \vec{w}^l, \vec{\mu}_l, C_l)$  ( $f_{il}$  can be initialized arbitrarily).

Note that we do not need to initialize other parameters because of the operation of the algorithm.

Let  $n$  be the origin node of link  $l$  and let link  $l$  be removed from the path of a session  $i$ . Session exit can cause removal of all links on its path. A link may also be removed from the path of a continuing session, if all receivers downstream leave.

1.  $\tau_i^n = \tau_i^n \setminus \{l\}$ ,  $\kappa_l = \kappa_l \setminus \{i\}$ ,
2. **estimate-link-control-parameter** $(\vec{f}^l, \vec{w}^l, \vec{\mu}_l, C_l)$ .

If capacity of a link changes anytime, then the link control parameter need simply be recomputed, with the new link capacity.

Subroutine **generate-forward-rate-packet** $(r_p, u_p)$  (**generate-feedback-rate-packet** $(r_p, u_p, \mu_p)$ ) simply generates a forward(feedback) rate packet with rate value equal to  $r_p$  and the state bit equal to  $u_p$  and minimum rate equal to  $\mu_p$ . Only a feedback rate packet has an entry for a minimum rate.

We describe **estimate-link-control-parameter** $(\vec{f}^l, \vec{w}^l, \vec{\mu}_l, C_l)$  below.

**estimate-link-control-parameter** $(\vec{f}^l, \vec{w}^l, \vec{\mu}_l, C_l)$  {

If  $\kappa_l = \phi$ ,

$\{\psi_l = C_l$   
end subroutine;}

If  $w_{il} = 1, \forall i \in \kappa_l$ ,



{ $\psi_l = C_l - \sum_{i \in \kappa_l} f_{il} + \max_{i \in \kappa_l} f_{il}$   
 While  $f_{il} \geq \psi_l$ , for some  $i \in \kappa_l$ ,  $w_{il} = 0$   
 If  $w_{il} = 1 \forall i \in \kappa_l$ , end subroutine;}

If  $\sum_{i:w_{il}=0} \hat{\mu}_{il} < C_l - \sum_{i:w_{il}=1} f_{il}$

{Set  $w_{il} = 0$  for all  $i \in \kappa_l$ .  
 If  $\sum_{i \in \kappa_l} \hat{\mu}_{il} = C_l$ ,  $\psi_l = 0$   
 else determine  $\psi_l$  s.t.  $\sum_{i \in \kappa_l} \psi_{il} = C_l$ ,<sup>‡</sup> where  $\psi_{il} = \max(\psi_l, \hat{\mu}_{il})$ .  
 end subroutine;}

If  $\sum_{i:w_{il}=0} \hat{\mu}_{il} = C_l - \sum_{i:w_{il}=1} f_{il}$  (minimum rate test)

{If there does not exist  $i \in \kappa_l$ , s.t.  $w_{il} = 1$  and  
 $f_{il} > \hat{\mu}_{il}$  then  
 {Set  $w_{il} = 0$  for all  $i \in \kappa_l$ .  
 $\psi_l = 0$   
 end subroutine;}

$\beta_l = \max_{i:f_{il} > \hat{\mu}_{il}, w_{il}=1} f_{il}$   
 Set  $w_{il} = 0 \forall i$  for which  $f_{il} \geq \beta_l$ .}

determine  $\psi_l$  s.t.  $\sum_{i:w_{il}=0} \psi_{il} = C_l - \sum_{i:w_{il}=1} f_{il}$ , where  $\psi_{il} = \max(\psi_l, \hat{\mu}_{il})$ .

If  $w_{il} = 1$ , and  $f_{il} > \psi_l$  for some  $i \in \kappa_l$ , then

{ set  $w_{il} = 0 \forall i \in \kappa_l$  s.t.  $f_{il} > \psi_l$ .  
 determine  $\psi_l$  s.t.  $\sum_{i:w_{il}=0} \psi_{il} = C_l - \sum_{i:w_{il}=1} f_{il}$ ,  
 where  $\psi_{il} = \max(\psi_l, \hat{\mu}_{il})$ .}

end subroutine; }

## 5 Allocation of Rates

In this section, we discuss the generic mechanisms for allocation of the fair rates, once they are computed.

---

<sup>‡</sup>If  $\sum_{i \in \kappa_l} \hat{\mu}_{il} < C_l$ , then there is a unique  $\psi_l$  which satisfies  $\sum_{i \in \kappa_l} \psi_{il} = C_l$ . If  $\sum_{i \in \kappa_l} \hat{\mu}_{il} = C_l$ , then  $\psi_l = 0$ , satisfies  $\sum_{i \in \kappa_l} \psi_{il} = C_l$ .

Once the fair rates are computed, the rate of a signal flow can be adjusted from within the network. Video gateways are placed throughout the network to transcode signal into a lower bit rate coding, using either the same coding format with different parameters or a different coding scheme altogether[31]. Source transmits at a rate equal to the maximum of the fair rates allocated to its receivers. Nodes know the maxmin fair session link rates in the outgoing links on account of the distributed computation. At forking points, video gateways transcode signal into a lower bit rate such that the rate in every link is equal to the maxmin fair session link rate, which is the maximum of the fair rates allocated to the session receivers downstream[31]. For instance, the maxmin fair rates of virtual sessions 1, 2, 3 are 4, 3.5, 3 respectively in Example 3.1.1. Virtual sessions 1, 2 belong to session 1 and virtual session 3 belongs to session 3. Source of session 1 transmits at rate 4. The video gateway at intermediate node  $I$  (Figure 2) transcodes session 1 signal to rate 3.5 for transmission through  $e_3$  and further downstream to  $u_2$ . Session 1 signal is transmitted at rate 4 units through  $e_2$  and  $e_4$ , though. In this way, heterogeneity is locally accommodated to fine tune the transmission rate to exactly match the fair share of link bandwidth allocated to receivers downstream. Rate adaptive video gateways [2] can be used for this purpose. Rate control is attained by employing more aggressive quantizers and by output frame rate control (dropping frames as necessary from the input stream to meet a given rate constraint). The latter attains fine tuning of the output bit rate. Active network architecture[33] provides a framework for deployment of rate adaptive video gateways within the network.

A second solution to the problem of fair allocation of rates is multirate transmission using hierarchical encoding. In this approach, an information stream is partitioned into a base layer, comprising the information needed to represent the lowest fidelity media and a number of enhancement layers. Some number of these enhancement layers are combined by the decoder with the base layer to recover a signal of incremental fidelity. An important characteristic of hierarchical encoding is that the layers are generally ordered in some manner. An enhancement layer yields useful information only when the base layer and the lower layers have been successfully decoded. As each layer is added, there is an improvement of quality of the received signal and additional bandwidth is needed to transport the combined stream. Hierarchical coding was first suggested for packet voice transmission[7] and subsequently several coding schemes have been proposed for video, e.g., [1], [17], [34], to

name a few. Some of these are amenable to tuning of layer bandwidths. Tuning of layer bandwidths become trivial if an *embedded* code is used. In an embedded code, any prefix of a valid codeword is a valid codeword and if the code is chosen appropriately, the prefix codeword corresponds to a lower quality version of the longer codeword. Hence one can trivially form a layered code from an embedded code by breaking up the longer code at arbitrary boundaries. Moreover, one can generate as many layers as desired and tune the rates of the different layers since the embedded code can be broken at arbitrary points. A low complexity video codec is presented in [24]. It uses PVH (progressive video with hybrid transform) code, an instance of an embedded code and is thus amenable to dynamic rate control. Embedded code has also been used in speech coding [7]. A pyramidal video coding scheme with two layers and a quality parameter to control the bit rate of the enhancement layer has been suggested in [8]. This idea may be extended to multilayer coding with flexible layer bandwidths. Tuning of layer bandwidths can be incorporated in many other codecs by using adaptive quantizers instead of fixed ones. If the source coder is programmable, and the source can modify its signals hierarchical structure, as in the cases discussed above, once the fair rates are computed, the source partitions its signal to form as many layers as there are distinct receiver rates and tunes the layer bandwidths to match the receiver rates. Each receiver is allotted layers as per its fair share of bandwidth. Total number of layers transmitted across a link is equal to the maximum number of layers allotted to the session receivers downstream. Layers are dropped or packets are replicated at forking points selectively. For instance, the source of session 1 in Example 3.1.1 generates a code with 2 layers. The base layer will have bandwidth equal to 3.5 units, and the enhancement layer will have bandwidth equal to .5 unit. Both layers are transmitted across  $e_1, e_2, e_4$ . Only the base layer is transmitted across  $e_3, e_5$ . Only base layer packets are replicated at intermediate node  $I$ , for transmission across  $e_2$  and  $e_3$ . The enhancement layer packets are not replicated at  $I$  as they are transmitted only across  $e_2$ . The source in Figure 1 will have a 4-layer encoding. Receiver  $u_1$  receives the first two layers,  $u_2$  receives all 4 of them,  $u_3$  receives the first 3 layers and the slowest receiver  $u_4$ , receives only the base layer. We would like to mention that layered coding is useful, if the nodes of the multicast tree are able to selectively forward flows to a specific branch. This mechanism is likely to be deployed in networks eventually. If network nodes do not have this ability, then transcoding is the only option.

A perfect match between the layer bandwidths and the maxmin fair receiver rates is not always feasible due to limitations on such parameters as the number of layers and the minimum bandwidth assigned to a layer. Although PVH codec of [24] is amenable to dynamic layer bandwidth adaptation, the implementations still have a fixed layout strategy. Besides, many hierarchical coding schemes do not generate embedded codes, e.g., perceptually weighted wavelets using hierarchical vector quantization with wavelet decomposition (WWHVQ) [36]. In certain video codes, substreams can be extracted to produce a specific range of resolutions only, e.g., 3D Subband Video Coding [34]. Certain coding schemes are particularly successful only when some a priori structure or hierarchy can be found in the problem [35]. In many cases, generating a layer requires a dedicated filter, and the number of filters employed by the source is fixed. Thus the number of layers is limited. Similarly, requirements for minimum packet size and limitations on packetization delay, may impose a lower bound on bandwidth assigned to a single layer and thus adversely affect layer granularity. The following approach can be adopted in cases of fixed or partially adaptive signal hierarchy. Allocate to the receivers as many layers as permitted by the computed fair rate. If the total bandwidth consumed by the layers allocated to a receiver is strictly less than the fair computed rate, allocate one more layer to the receiver and let the network drop a certain portion of packets of the last layer at a forking point. For instance, if the source of session 1 in Example 3.1.1 can transmit unit bandwidth layers only, then initially receiver  $u_1$  will be allotted 4 layers and receiver  $u_2$  3 layers only. Receiver  $u_2$  will be allotted one more layer in the second stage (4 layers in all), because its fair rate is 3.5 units and 3 layers consume 3 units of bandwidth. Intermediate node  $I$  should not replicate 50% of packets of layer 4. All layer 4 packets are transmitted across  $e_2$  and only 50% of them are transmitted across  $e_3$ . So 50% of layer 4 packets should not be replicated at node  $I$ . For certain coding schemes, packet loss causes a graceful degradation in signal quality. For example, in [17] signal exhibits a reasonable enhancement in quality above the base layer for 10% enhancement layer packet loss. This transition is gradual up until 100% enhancement layer packet loss. This mechanism will be useful in these coding schemes.

We would like to comment on our assumption that all multicast packets of the same session move along the same tree. Different multicast layers are perceived as different multicast groups and trees for different groups may be

completely different in general. However, trees for different multicast layers of the same session will remain the same if source rooted trees are used, as all these layers (multicast groups) have the same source. Besides, trees for different multicast layers of the same session should not differ very much, as that would complicate reconstruction of information at the receiver. For example, if we consider video transmission, and different layer packets of the same session traverse along different multicast trees, then different layer packets for the same frame may arrive at a receiver at different times, and frame reconstruction will involve a lot of packet reordering. This may incur an unacceptable delay jitter. Thus it may be a good idea to use source rooted trees in this case. Many video coders make the same assumption, e.g., [8].

## 6 Conclusion

Summarizing, we have presented a mathematical framework that can model the fair allocation of bandwidth in the multicast scenario, while taking care of both intra-session and inter-session fairness. This framework can model fairness in both internet and ATM like networks. We have presented an algorithm for computation of the maxmin fair rates in arbitrary multicast networks and a framework for distributed implementation of the same. The algorithm takes care of minimum rate requirements, which are often present in ATM ABR multicast scenario. This algorithm can be used to make maxmin fair rate allocation, in presence of bandwidth adaptive hierarchical coding or when the network has certain provisions like transcoding option or selective packet replication. Several well known algorithms like the algorithm for computation of maxmin fair rates in arbitrary unicast network and algorithm for computation of maxmin fair rates in multicast network with the restriction that all receivers of the same session must receive service at the same rate are in fact special case of ours. We have also presented another distributed algorithm for computation of maxmin fair rates in a multirate multicast network which is more suited to asynchronous operation, i.e., when sessions enter and exit at different times. There exists an array of related intellectually challenging unsolved research problems.

Firstly, we have assumed that network has certain provisions, but that may not be the case. For example, transcoding places additional computational and administrative burden on the network. It may be possible to de-

ploy transcoding gateways at strategic locations in the network but a system that requires transcoding at arbitrary bottleneck links may not be computationally and financially viable. For transcoding, internal nodes must perform intensive computational tasks and may need to process arbitrary coding algorithms. Transcoding process increases end to end delay and cannot be applied to a secure communication without entrusting the network with the encryption key. Depending on the security risks, this may be totally unacceptable. In these cases, network heterogeneity can be countered through hierarchical encoding only. As discussed before, hierarchical signal structure may be predetermined and layer granularity may be lower bounded. Network may not have selective replication provisions at congestion points. As a consequence, congestion will affect all the layers and the signal quality will be very poor. Besides trusting the network to drop a certain proportion of the highest layer packets may produce perceptually annoying distortions in certain coding schemes. If the highest layer, employs a differential coding scheme, then even a small percentage loss of packets may garble the entire information in the final layer. In all these cases, “partial” subscription to a layer is useless and receivers can only subscribe to layers fully. The consequence is that *any* rate vector in the current feasible set can not be allocated. Rather, only a discrete subset of rate vectors in the feasible set can actually be allocated on account of the constraints on the signal structure. Fairness in a discrete feasible set is vastly different from fairness in our current feasible set. It opens up many new research problems. We address those in [30].

We have discussed the generic mechanisms for fair rate allocation in presence of certain provisions like transcoding options and/or adaptive hierarchical encoding in Section 5, but we did not describe the exact protocol format for the purpose. The detailed protocol would depend on the particular choice (transcoding or adaptive hierarchical encoding). Developing the detailed protocols for the provisions described in Section 5 would be interesting from an implementation point of view. We have also not described the exact message exchange format for distributed computation of the maxmin fair rates, but have suggested the general framework. Developing the exact protocol format will have implementational significance. Besides, while describing both the distributed algorithms, we had implicitly assumed that the control messages (e.g., rate packets, probe messages, saturation, unsaturation messages) are never lost. However, in internet like lossy networks, control messages may be lost. We need to devise retransmission and timeout strategies. This is a

topic of future investigation.

## A Proof of Bottleneck Lemma:

**Proof of Lemma 1(Bottleneck Lemma):** Let a virtual session  $s$  not have a bottleneck link under a feasible rate vector  $\vec{r}$ . If  $s$  traverses a link  $l$ , it has one of the following three properties.

1. Link  $l$  has some unused capacity.
2. Rate of some other virtual session of the same session as  $s$  traversing  $l$  has greater rate than that of  $s$ , i.e., bandwidth consumed by the session of  $s$  in link  $l$  is greater than the rate of  $s$ .
3. A session  $i$  traversing link  $l$  consumes greater bandwidth than  $s$  in link  $l$ . This bandwidth is greater than the minimum rate of session  $i$  in link  $l$  ( $\mu_{il}$ ). Virtual session  $s$  does not belong to session  $i$ .

If all links on the path of  $s$  satisfy the first two conditions, then the rate of  $s$  can be increased without decreasing that of any other virtual sessions. If some links on the path of  $s$  satisfy only the last property, then to increase the rate of virtual session  $s$ , we may need to decrease the rate of some virtual sessions having greater rate than that of  $s$  though. However we can still increase the rate of virtual session  $s$ , without decreasing that of any other virtual session having rate less than that of  $s$  and still maintain feasibility. Thus  $\vec{r}$  is not a maxmin fair rate vector.

Let  $\vec{r}^1$  be a feasible rate vector such that all virtual sessions have a bottleneck link. Consider any other feasible rate vector  $\vec{r}^2$ . Let there exist a virtual session  $s$  such that  $r_s^2 > r_s^1$ . Let  $l$  be a bottleneck link for virtual session  $s$ .  $\lambda_{\chi(s),l}^2 \geq r_s^2 > r_s^1 = \lambda_{\chi(s),l}^1$ . The last equality follows from the property of a bottleneck link. Since  $l$  is a bottleneck link w.r.t. virtual session  $s$ , its capacity is fully utilized, i.e.,  $\sum_{i \in n(l)} \lambda_{il}^1 = C_l$ . From the feasibility of  $\vec{r}^2$ ,  $\sum_{i \in n(l)} \lambda_{il}^2 \leq C_l$ . Since  $\lambda_{\chi(s),l}^2 > \lambda_{\chi(s),l}^1$ , and  $\chi(s) \in n(l)$  (since virtual session  $s$  traverses through link  $l$ ), it follows that there exists a session  $j$  such that  $\lambda_{jl}^2 < \lambda_{jl}^1$ . From feasibility of  $\vec{r}^2$ ,  $\lambda_{jl}^2 \geq \mu_{jl}$ . It follows that  $\lambda_{jl}^1 > \mu_{jl}$ . There exists virtual session  $t$  such that  $t \in m(j, l)$ ,  $r_t^1 = \lambda_{jl}^1$ . It follows that  $r_t^1 > \mu_{\chi(t)l}$ . It follows from the last condition for a link to be bottleneck w.r.t. a virtual session that  $r_t^1 \leq r_s^1$ . Now  $r_t^2 \leq \lambda_{jl}^2 < \lambda_{jl}^1 = r_t^1$ . The first inequality

follows since  $\chi(t) = j$ . The second inequality and the last equality have been argued before. Thus it follows that  $r_s^2 > r_s^1$  implies there exists virtual session  $t$  such that  $r_t^2 < r_t^1 \leq r_s^1$ . Hence  $\vec{r}^1$  is a maxmin fair rate vector.  $\square$

## B Proof of correctness for the algorithm for computation of Maxmin fair rates

We prove that the algorithm for computation of maxmin fair rates presented in Section 3.1 indeed yields a maxmin fair rate vector upon termination (Theorem 1) and the algorithm terminates in  $M$  iterations (Theorem 2), where  $M$  is the number of virtual sessions. We assume that the set of feasible rate vectors is nonempty. We outline the proof of Theorem 1 as follows. We first show that the link control parameters increase (do not decrease, to be precise) with every iteration (Lemma 3). It follows that the virtual session rates and the session rates do not decrease in subsequent iterations (Lemma 4). Using this, we show that the rate allocation at the end of the every iteration is feasible (Lemma 6). Next we show that, if a virtual session saturates in the  $k$ th iteration, it has a bottleneck link in all subsequent iterations (Lemma 7). Since the algorithm terminates only when all virtual sessions saturate, all virtual sessions have a bottleneck link when the algorithm terminates. The rate allocation upon termination is also feasible by Lemma 6. Thus maxmin fairness of the rate allocation upon termination follows from the Bottleneck Lemma.

**Lemma 3** *If  $k \geq 1$  and the algorithm has not terminated in  $k - 1$  iterations,  $\eta_l(k) \geq \eta_l(k - 1) \forall k \geq 1$ .*

**Remark:** The intuition behind the result is as follows. Ignore the minimum rate constraints for the time being. Link control parameter in an iteration is the residual capacity per unsaturated session. The bandwidth assigned to every unsaturated session traversing the link is at most equal to this link control parameter. It may be less on account of bandwidth constraints on other links, though. If no new session is saturated this iteration, then the number of unsaturated sessions and the residual bandwidth remains the same in the next iteration. So the link control parameter remains the same as well. If a new session is saturated, then the residual capacity in the next



iteration decreases by its bandwidth on the link, but this bandwidth is at most by the link control parameter. So the link control parameter in the next iteration does not decrease with respect to that in the current iteration. The formal proof follows. The formal proof does not ignore the minimum rate constraints.

**Proof of Lemma 3:** We will prove by induction. Let  $k = 1$ . The algorithm can not terminate in 0 iterations.  $S(0) \neq \phi$ ,  $F_l(0) = 0$ . If  $\Xi_l(0) = \phi$ , then  $\eta_l(1) = \eta_l(0) = 0$ , and the lemma holds for link  $l$  and iteration 1. Let  $\Xi_l(0) \neq \phi$ . Since the feasible set of rate vectors is nonempty,  $\sum_{i \in \Xi_l(0)} \mu_{il} \leq C_l$ .  $\mu_{il} \geq 0$ , for every session  $i$  and link  $l$ . Thus  $\theta = 0$ , satisfies the inequality  $\sum_{i \in \Xi_l(0)} \max(\theta, \mu_{il}) \leq C_l$ . Clearly  $\eta_l(1)$  is the maximum possible  $\theta$  which satisfies the above inequality. Hence  $\eta_l(1) \geq 0 = \eta_l(0)$ . The equality follows from the initialization of  $\eta_l(0)$ . Thus the lemma holds for  $k = 1$ .

Let the lemma hold for iterations  $1, \dots, k$  and  $S(k) \neq \phi$ . We will show that the lemma holds for the  $k+1$ th iteration. If  $\Xi_l(k) = \phi$ , then  $\eta_l(k+1) = \eta_l(k)$ . Let  $\Xi_l(k) \neq \phi$ . Consider any virtual session  $s$  traversing through link  $l$ , i.e.,  $\chi(s) \in n(l)$ . If  $s \in S(k-1)$ ,  $r_s(k) \leq \eta_{\chi(s)l}(k) = \max(\eta_l(k), \mu_{\chi(s)l})$ . Let  $s \notin S(k-1)$ . Thus  $s \in S(t) \setminus S(t+1)$ ,  $0 \leq t < k-1$ .  $r_s(t+1) \leq \max(\eta_l(t+1), \mu_{\chi(s)l}) \leq \max(\eta_l(k), \mu_{\chi(s)l})$ , since  $t+1 < k$ , and  $\eta_l(0) \leq \eta_l(1) \leq \dots \leq \eta_l(k)$  by induction hypothesis. Since  $S(k-1) \subseteq \dots \subseteq S(t+1)$  ( $t+1 < k$  and  $S(p+1) \subseteq S(p)$ ,  $\forall p$ ),  $s \notin S(t+1), \dots, S(k-1)$ . Thus  $r_s(k) = \dots = r_s(t+1) \leq \max(\eta_l(k), \mu_{\chi(s)l})$ .

$$r_s(k) \leq \max(\eta_l(k), \mu_{\chi(s)l}) \text{ for all virtual sessions } s \quad (2)$$

$$\text{Thus } \lambda_{il}(k) \leq \max(\eta_l(k), \mu_{il}) \quad (3)$$

$$\begin{aligned} F_l(k) - F_l(k-1) &= \sum_{i \in \Xi_l(k-1) \setminus \Xi_l(k)} \lambda_{il}(k) \\ &\leq \sum_{i \in \Xi_l(k-1) \setminus \Xi_l(k)} \max(\eta_l(k), \mu_{il}) \quad (\text{from (3)}) \\ F_l(k) + \sum_{i \in \Xi_l(k)} \max(\eta_l(k), \mu_{il}) &\leq F_l(k-1) + \sum_{i \in \Xi_l(k-1) \setminus \Xi_l(k)} \max(\eta_l(k), \mu_{il}) + \\ &\quad \sum_{i \in \Xi_l(k)} \max(\eta_l(k), \mu_{il}) \\ &= F_l(k-1) + \sum_{i \in \Xi_l(k-1)} \max(\eta_l(k), \mu_{il}) \\ &\quad (\text{since } \Xi_l(k) \subseteq \Xi_l(k-1) \text{ as } S(k) \subseteq S(k-1)) \end{aligned}$$

$$= C_l$$

Thus  $\theta = \eta_l(k)$  satisfies the inequality  $F_l(k) + \sum_{i \in \Xi_l(k)} \max(\theta, \mu_{il}) \leq C_l$ . Clearly  $\eta_l(k+1)$  is the maximum possible value of  $\theta$  which satisfies the inequality and hence there exists a  $\eta_l(k+1)$  and  $\eta_l(k+1) \geq \eta_l(k)$ . Hence the result follows from induction.  $\square$

**Lemma 4**  $r_s(k+1) \geq r_s(k)$  if  $k \geq 0$  for all virtual sessions  $s$ .  $\lambda_{il}(k+1) \geq \lambda_{il}(k)$  if  $k \geq 1$  for all sessions  $i$  and links  $l$ .

**Remark:** We present the intuition, ignoring the minimum rate constraints. The formal proof will consider them. The rate of an unsaturated virtual session is equal to the minimum link control parameter on its path. Link control parameter increases (does not decrease) with the iteration (Lemma 3). Thus the result follows for unsaturated virtual sessions. The rate of a saturated virtual session does not change in subsequent iterations. Thus the result follows for virtual sessions. Session rates are nondecreasing because a session rate on a link is equal to the rate of the virtual sessions of the session traversing the link. The formal proof follows.

**Proof of Lemma 4:** Let  $k = 0$ .  $s \in S(0)$  for all virtual sessions  $s$ .  $r_s(1) = \min_{l \in L_s} \eta_{\chi(s)l}(1)$ .  $\eta_{\chi(s)l}(1) = \max(\eta_l(1), \mu_{\chi(s)l}) \geq \mu_{\chi(s)l} \geq \mu_s$ . The last inequality follows from the fact that  $s \in m(\chi(s), l)$ . Thus  $r_s(1) \geq \mu_s = r_s(0)$ . Thus the result holds for  $k = 0$ . Let  $k \geq 1$ . If  $s \notin S(k)$ ,  $r_s(k+1) = r_s(k)$ . If  $s \in S(k)$ ,  $r_s(k+1) = \min_{l \in L_s} \eta_{\chi(s)l}(k+1)$ .  $\eta_{\chi(s)l}(k+1) = \max(\eta_l(k+1), \mu_{\chi(s)l})$ .  $\eta_l(k+1) \geq \eta_l(k)$ . Thus  $\eta_{\chi(s)l}(k+1) \geq (\eta_l(k), \mu_{\chi(s)l}) = \eta_{\chi(s)l}(k)$ . Thus  $r_s(k+1) \geq \min_{l \in L_s} \eta_{\chi(s)l}(k)$ . Since  $s \in S(k)$ , and  $S(k-1) \supseteq S(k)$ ,  $s \in S(k-1)$ . Thus  $r_s(k) = \min_{l \in L_s} \eta_{\chi(s)l}(k)$ . Hence  $r_s(k+1) \geq r_s(k)$ . The second part of the lemma follows from the first and the fact that  $\lambda_{il}(k) = \max_{s \in m(i,l)} r_s(k)$   $\square$

**Lemma 5**

$$\begin{aligned} r_s(k) &\leq \eta_{\chi(s)l}(k) \text{ for all virtual sessions } s, \text{ links } l \in L_s \text{ and } k \geq 0 \\ \text{Thus } \lambda_{il}(k) &\leq \eta_{il}(k) \text{ for all sessions } i, \text{ links } l \text{ and } k > 0 \end{aligned} \quad (5)$$

**Remark:** The intuition is as follows. Rate of an unsaturated virtual session  $s$  is equal to the minimum value of its session link control parameter  $\eta_{\chi(s)l}(k)$

on its path. So the result holds for an unsaturated virtual session. If a virtual session is saturated in the  $k$ th iteration, its rate is frozen at the value of the minimum session link control parameter in the  $k$ th iteration on its path. Session link control parameter increases (nondecreasing) in subsequent iteration. Thus the result follows. The result for session bandwidths follow similarly. We will use this lemma in proofs of feasibility of rate allocations (Lemma 6), the fact that every saturated virtual session has a bottleneck link (Lemma 7) and the fact that the algorithm terminates in finite number of iterations (Theorem 2). The formal proof follows.

**Proof of Lemma 5:** We prove (4) by induction.  $r_s(0) = \mu_s \leq \max(\eta_l(0), \mu_{\chi(s)l})$   $\forall l \in L_s$ . The last inequality follows from the fact that  $\mu_{\chi(s)l} = \max_{j \in m(\chi(s), l)} \mu_j$  and  $s \in m(\chi(s), l)$ , since  $l \in L_s$ . Thus (4) holds for  $k = 0$ .

Let (4) hold for  $k \geq 0$ . If  $s \in S(k)$ ,  $r_s(k+1) \leq \eta_{\chi(s)l}(k+1)$ . If  $s \notin S(k)$ ,

$$\begin{aligned} r_s(k+1) &= r_s(k) \\ &\leq \eta_{\chi(s)l}(k) \text{ (from induction hypothesis)} \\ &= \max(\eta_l(k), \mu_{\chi(s)l}) \\ &\leq \max(\eta_l(k+1), \mu_{\chi(s)l}) \text{ ( } \eta_l(k) \leq \eta_l(k+1) \text{ by Lemma 3)} \\ &\leq \eta_{\chi(s)l}(k+1) \end{aligned}$$

Thus (4) holds for  $k+1$ . Thus (4) holds for all  $k \geq 0$  by induction. (5) follows from (4) and the definition of  $\lambda_{il}(k)$ .  $\square$

**Lemma 6** *The rate allocation at the end of the  $k$ th iteration is feasible,  $k \geq 0$ .*

**Remark:** Intuitive reasoning for the result is as follows. Rate of a virtual session at the 0th iteration is its minimum rate. Rate of a virtual session do not decrease in subsequent iterations. Thus minimum rate constraints are satisfied. Capacity constraints hold because no unsaturated session is offered a rate more than its session link control parameter in any link. Session link control parameters are computed such that the sum of the session link control parameters for unsaturated sessions and bandwidths for saturated sessions equal the link capacity for every link. The formal proof follows.

**Proof of Lemma 6:** We prove by induction. We first prove the lemma for  $k = 0$ . Since the set of feasible rate vectors is nonempty, a rate allocation with rate of each virtual session  $s$  equal to its minimum rate satisfies the capacity

constraints. Thus  $\vec{r}(0)$  satisfies the capacity constraints. Since  $r_s(0) = \mu_s$ ,  $\vec{r}(0)$  satisfies the minimum rate requirements. Thus  $\vec{r}(0)$  is feasible and the lemma holds for  $k = 0$ .

Let the rate allocation at the end of the  $k$ th iteration,  $\vec{r}(k)$  be feasible. Consider the  $k + 1$ th iteration.  $r_s(k + 1) \geq r_s(k) \geq \mu_s$ . The first inequality follows from Lemma 4 and the last from the feasibility of  $\vec{r}(k)$ . Thus  $r_s(k + 1) \geq \mu_s$ , for all virtual sessions  $s$ .

$$\begin{aligned} r_s(k + 1) &= r_s(k) \text{ (if } \chi(s) \in n(l) \setminus \Xi_l(k) \text{ since then } s \notin S(k)) \\ \text{Thus } \lambda_{il}(k + 1) &= \max_{s \in m(i,l)} r_s(k) \text{ (if } i \in n(l) \setminus \Xi_l(k)) \\ &= \lambda_{il}(k) \text{ (if } i \in n(l) \setminus \Xi_l(k)) \end{aligned} \quad (6)$$

If  $\Xi_l(k) = \phi$ , from (6),  $\lambda_{il}(k + 1) = \lambda_{il}(k) \forall i \in n(l)$ .

$$\begin{aligned} \sum_{i \in n(l)} \lambda_{il}(k + 1) &= \sum_{i \in n(l)} \lambda_{il}(k) \\ &\leq C_l \text{ (from the feasibility of } \vec{r}(k)) \end{aligned}$$

If  $\Xi_l(k) \neq \phi$ ,

$$\begin{aligned} \sum_{i \in n(l)} \lambda_{il}(k + 1) &= \sum_{i \in n(l) \setminus \Xi_l(k)} \lambda_{il}(k + 1) + \sum_{i \in \Xi_l(k)} \lambda_{il}(k + 1) \\ &\leq \sum_{i \in n(l) \setminus \Xi_l(k)} \lambda_{il}(k) + \sum_{i \in \Xi_l(k)} \max(\eta_l(k + 1), \mu_{il}) \\ &\quad \text{(from (5) of Lemma 5 and (6))} \\ &= F_l(k) + \sum_{i \in \Xi_l(k)} \max(\eta_l(k + 1), \mu_{il}) \\ &= C_l \end{aligned}$$

Since link  $l$  was chosen arbitrarily, it follows that the rate vector at the end of iteration  $k + 1$ ,  $\vec{r}(k + 1)$  satisfies the capacity condition for every link  $l$ . Hence  $\vec{r}(k + 1)$  is feasible. Thus the lemma follows by induction.  $\square$

**Lemma 7** *Let  $s \in S(k - 1) \setminus S(k)$ . Let  $t \geq k$  and  $S(t - 1) \neq \phi$ , i.e., the algorithm does not terminate in  $t - 1$  iterations. Then virtual session  $s$  has a bottleneck link, under rate vector  $\vec{r}(t)$ .*

**Remark:** Intuitively the result can be argued as follows. Ignore the minimum rate constraints for time being. If a virtual session  $s$  saturates, then it traverses through a link  $l$  whose capacity is fully utilized. Capacity of the same link remains utilized in all subsequent iterations because session rates do not decrease. Thus the first bottleneck condition holds in all iterations. Actually, rates of sessions traversing the link do not change in subsequent iterations, because they can not increase in view of the full utilization of the capacity, and they can not decrease in any event. The rate of the particular virtual session concerned does not change any further. Thus, since the second bottleneck condition holds in the iteration in which the virtual session saturate (by condition for saturation), it holds in all subsequent iterations. We argue the third condition as follows. It can be shown that if  $s$  saturates in the  $k$ th iteration, then its rate is equal to the link control parameter of  $l$  in the  $k$ th iteration, and rate of any other virtual session traversing  $l$  in any subsequent iteration is upper bounded by the link control parameter of  $l$  in the  $k$ th iteration. Thus the rate of any other virtual session traversing  $l$  does not exceed that of  $s$  in any subsequent iteration. Thus the third bottleneck condition holds. The formal proof argues the lemma in a more general case, with the minimum rate constraints.

**Proof of Lemma 7:** Let  $s \in S(k-1) \setminus S(k)$ , i.e., virtual session  $s$  becomes saturated at the end of the  $k$ th iteration. Thus there exists link  $l \in L_s$  such that

$$\sum_{i \in n(l)} \lambda_{il}(k) = C_l \quad (7)$$

$$\text{and } r_s(k) = \lambda_{\chi(s)l}(k) \quad (8)$$

By Lemma 5,  $r_s(k) \leq \eta_{\chi(s)l}(k)$ . Let  $r_s(k) < \eta_{\chi(s)l}(k)$ .

$$\sum_{i \in n(l)} \lambda_{il}(k) = \sum_{i \in n(l) \setminus \Xi_l(k-1)} \lambda_{il}(k) + \sum_{i \in \Xi_l(k-1) \setminus \chi(s)} \lambda_{il}(k) + \lambda_{\chi(s)l}(k) \quad (9)$$

$$\lambda_{il}(k) = \lambda_{il}(k-1) \text{ if } i \in n(l) \setminus \Xi_l(k-1) \text{ (argued in (6))} \quad (10)$$

$$\lambda_{il}(k) \leq \max(\eta_l(k), \mu_{il}) \forall i \in n(l) \text{ (from (5) of Lemma 5)} \quad (11)$$

$$\begin{aligned} \lambda_{\chi(s)l}(k) &= r_s(k) \\ &< \eta_{\chi(s)l}(k) \\ &= \max(\eta_l(k), \mu_{\chi(s)l}) \end{aligned} \quad (12)$$

$$\begin{aligned}
\sum_{i \in n(l)} \lambda_{il}(k) &< \sum_{i \in n(l) \setminus \Xi_l(k-1)} \lambda_{il}(k-1) + \sum_{i \in \Xi_l(k-1)} \max(\eta_l(k), \mu_{il}) \quad (13) \\
&= F_l(k-1) + \sum_{i \in \Xi_l(k-1)} \max(\eta_l(k), \mu_{il}) \\
&= C_l \text{ (since } s \in S(k-1), l \in L_s \text{ implying } \Xi_l(k-1) \neq \emptyset) \quad (14)
\end{aligned}$$

The breakup in (9) follows from the fact that  $\Xi_l(k-1) \subseteq n(l)$  and  $\chi(s) \in \Xi_l(k-1)$  as  $l \in L_s$ ,  $s \in S(k-1)$  and thus  $m(\chi(s), l) \cap S(k-1) \supseteq \{s\}$ . (13) follows from (9), (10) and (12). (14) contradicts (7). Thus

$$r_s(k) = \eta_{\chi(s)l}(k) \quad (15)$$

Consider the  $t$  th iteration,  $t \geq k$ .

$$\begin{aligned}
\sum_{i \in n(l)} \lambda_{il}(t) &\geq \sum_{i \in n(l)} \lambda_{il}(k) \text{ (since } \lambda_{il}(t) \geq \lambda_{il}(k), t \geq k \text{ from Lemma 4)} \\
&= C_l \\
\sum_{i \in n(l)} \lambda_{il}(t) &\leq C_l \text{ (from feasibility of } \vec{r}(t)) \\
\text{Thus } \sum_{i \in n(l)} \lambda_{il}(t) &= C_l \quad (16)
\end{aligned}$$

Since  $\lambda_{i_1}(t) \geq \lambda_{i_1}(k)$ , for all sessions  $i$ , links  $l_1$ , if  $\lambda_{il}(t) > \lambda_{il}(k)$ , for some  $i \in n(l)$ ,  $\sum_{i \in n(l)} \lambda_{il}(t) > \sum_{i \in n(l)} \lambda_{il}(k)$ . Thus from (7)  $\sum_{i \in n(l)} \lambda_{il}(t) > C_l$ . This contradicts the feasibility of  $\vec{r}(t)$  ( $\vec{r}(t)$  is feasible by Lemma 6). Thus

$$\lambda_{il}(t) = \lambda_{il}(k) \quad \forall i \in n(l) \quad (17)$$

Since  $s \notin S(k)$ ,  $r_s(t) = r_s(k)$ ,  $\forall t \geq k$ .  $r_s(k) = \lambda_{\chi(s)l}(k)$  by (8).  $l \in L_s$  and thus  $\chi(s) \in n(l)$ . Thus from (17)

$$r_s(t) = \lambda_{\chi(s)l}(t) \quad (18)$$

Let  $j$  be a virtual session traversing through link  $l$  and  $r_j(t) > r_s(t)$ .

$$\begin{aligned}
\max(\eta_l(k), \mu_{\chi(j)l}) &\geq \lambda_{\chi(j)l}(k) \text{ (from (5) of Lemma 5)} \\
&= \lambda_{\chi(j)l}(t) \text{ (from (17))} \\
&\geq r_j(t) \quad (19)
\end{aligned}$$

$$\begin{aligned}
 &> r_s(t) \text{ (by assumption)} \\
 &\geq r_s(k) \text{ (by Lemma 4 since } t \geq k) \\
 &= \max(\eta_l(k), \mu_{\chi(s)l}) \text{ (by (15))}
 \end{aligned}$$

$$\text{Thus } \max(\eta_l(k), \mu_{\chi(j)l}) = \mu_{\chi(j)l} \quad (20)$$

$$\begin{aligned}
 \text{Thus } r_j(t) &\leq \mu_{\chi(j)l} \text{ if } r_j(t) > r_s(t) \text{ and } l \in L_j \quad (21) \\
 &\text{(from (19) and (20))}
 \end{aligned}$$

From (16), (18) and (21), link  $l$  is bottlenecked w.r.t. virtual session  $s$  under rate vector  $\vec{r}(t)$ .  $\square$

**Proof of Theorem 1:** If the algorithm terminates in  $k$  iterations,  $S(k) = \phi$ . Since  $S(0)$  is the set of all virtual sessions, for every virtual session  $s$ , there exists a  $t \leq k$ , such that  $s \in S(t-1) \setminus S(t)$ ,  $t \leq k$ . Thus by Lemma 7 virtual session  $s$  has a bottleneck link  $l \in L_s$ , under rate vector  $\vec{r}(k)$ .  $\vec{r}(k)$  is a feasible rate vector by Lemma 6. Thus  $\vec{r}(k)$  is max-min fair by Lemma 1.  $\square$

Now we prove that the algorithm terminates in at most  $M$  iterations (Theorem 2), where  $M$  is the number of virtual sessions. The proof shows that a new virtual session saturates every iteration. It looks at the link  $l_{\min}$  which attains the minimum link control parameter, amongst all those which carry at least one unsaturated virtual session. It looks at that unsaturated session  $i_{\min}$  which has the minimum rate requirement (minimum  $\mu_{il}$ ) amongst all the unsaturated sessions traversing through link  $l_{\min}$  and argues that all unsaturated virtual sessions of  $i_{\min}$  saturates in the current iteration. The result follows. Refer to the formal proof for details.

**Proof of Theorem 2:** It is sufficient to prove that  $S(k)$  is a proper subset of  $S(k-1)$ ,  $\forall k$  s.t.  $S(k-1) \neq \phi$  and  $k \geq 1$ . Since  $|S(0)| = M$ , the result follows. Let  $l_{\min} = \arg \min_{l \in \cup_{s \in S(k-1)} L_s} \eta_l(k)$ .  $l_{\min}$  is well defined as  $S(k-1) \neq \phi$  (If more than one links attain the minimum, choose any one of them as  $l_{\min}$ ). Since  $l_{\min} \in \cup_{s \in S(k-1)} L_s$ , at least one virtual session in  $S(k-1)$  (unsaturated virtual session) traverses through link  $l_{\min}$ . Thus  $\Xi_{l_{\min}}(k-1) \neq \phi$ . Thus  $\eta_{l_{\min}}(k)$  is the maximum  $\theta$  which satisfies  $F_{l_{\min}}(k-1) + \sum_{i \in \Xi_{l_{\min}}(k-1)} \max(\theta, \mu_{il_{\min}}) \leq C_{l_{\min}}$  and observe that

$$F_{l_{\min}}(k-1) + \sum_{j \in \Xi_{l_{\min}}(k-1)} \max \left( \left( \min_{i \in \Xi_{l_{\min}}(k-1)} \mu_{il_{\min}} \right), \mu_{jl_{\min}} \right) \leq F_{l_{\min}}(k-1) +$$

$$\sum_{i \in \Xi_{l_{\min}}(k-1)} \max(\theta, \mu_{jl_{\min}}) \forall \theta$$

since  $\min_{i \in \Xi_{l_{\min}}(k-1)} \mu_{il_{\min}} \leq \mu_{jl_{\min}} \forall j \in \Xi_{l_{\min}}(k-1)$ . Thus

$$\eta_{l_{\min}}(k) \geq \min_{i \in \Xi_{l_{\min}}(k-1)} \mu_{il_{\min}} \quad (22)$$

Let  $i_{\min} = \arg \min_{i \in \Xi_{l_{\min}}(k-1)} \mu_{il_{\min}}$ .  $i_{\min}$  is well defined as  $\Xi_{l_{\min}} \neq \phi$ .

$$\begin{aligned} \eta_{\chi(s)l_{\min}}(k) &= \max(\eta_{l_{\min}}(k), \mu_{i_{\min}l_{\min}}) \forall s \in m(i_{\min}, l_{\min}) \cap S(k-1) \\ &= \eta_{l_{\min}}(k) \text{ (from (22) and the definition of } i_{\min}) \end{aligned} \quad (23)$$

Consider a  $s \in m(i_{\min}, l_{\min}) \cap S(k-1)$  ( $m(i_{\min}, l_{\min}) \cap S(k-1) \neq \phi$  as  $i_{\min} \in \Xi_{l_{\min}}(k-1)$ ).

$$\begin{aligned} \eta_{\chi(s)l_{\min}}(k) &= \eta_{l_{\min}}(k) \text{ (from (23))} \\ &\leq \eta_l(k) \forall l \in L_s \text{ (from the definition of } l_{\min} \text{ and since } s \in S(k-1)) \\ &\leq \eta_{\chi(s)l}(k) \forall l \in L_s \end{aligned}$$

$$\begin{aligned} \text{Thus } r_s(k) &= \eta_{\chi(s)l_{\min}}(k) \text{ (since } s \in S(k-1), r_s(k) = \min_{l \in L_s} \eta_{\chi(s)l}(k)) \\ &= \max(\eta_{l_{\min}}(k), \mu_{\chi(s)l_{\min}}) \\ &\geq \lambda_{\chi(s)l_{\min}}(k) \text{ (from (5) of Lemma 5)} \end{aligned} \quad (24)$$

$$r_s(k) \leq \lambda_{\chi(s)l_{\min}}(k)$$

$$\text{Thus } r_s(k) = \lambda_{\chi(s)l_{\min}}(k) \quad (25)$$

Consider any session  $i \in \Xi_{l_{\min}}(k-1)$ . Let  $\eta_{il_{\min}}(k) = \mu_{il_{\min}}$ .

$$\begin{aligned} \lambda_{il_{\min}}(k) &= \max_{s \in m(i, l_{\min})} r_s(k) \\ &\geq \max_{s \in m(i, l_{\min})} \mu_s \text{ (from feasibility of } \vec{r}(k)) \\ &= \mu_{il_{\min}} \\ &= \eta_{il_{\min}}(k) \text{ (from hypothesis)} \\ \lambda_{il_{\min}}(k) &\leq \eta_{il_{\min}}(k) \text{ (from (5) of Lemma 5)} \\ \text{Thus } \lambda_{il_{\min}}(k) &= \eta_{il_{\min}}(k) \end{aligned} \quad (26)$$

Now let  $\eta_{il_{\min}}(k) = \eta_{l_{\min}}(k)$ .  $r_j(k) = \eta_{il_{\min}}(k)$ ,  $\forall j \in m(i, l_{\min}) \cap S(k-1)$  ( $m(i, l_{\min}) \cap S(k-1) \neq \phi$  as  $i \in \Xi_{l_{\min}}(k-1)$ ). The reasoning is exactly the



same as that behind (24).  $\lambda_{il_{\min}}(k) = \max_{j \in m(i, l_{\min})} r_j(k) \geq \eta_{il_{\min}}(k)$ , since  $m(i, l_{\min}) \cap S(k-1) \neq \phi$ . From (5) of Lemma 5,

$$\lambda_{il_{\min}}(k) = \eta_{il_{\min}}(k) \quad (27)$$

From (26) and (27)

$$\lambda_{il_{\min}}(k) = \eta_{il_{\min}}(k) \quad \forall i \in \Xi_l(k-1) \text{ (from (26) and (27))} \quad (28)$$

$$\begin{aligned} \lambda_{il_{\min}}(k) &= \lambda_{il_{\min}}(k-1) \text{ if } i \in n(l_{\min}) \setminus \Xi_{l_{\min}}(k-1) \text{ (from (10))} \\ \sum_{i \in n(l_{\min}) \setminus \Xi_{l_{\min}}(k-1)} \lambda_{il_{\min}}(k) &= \sum_{i \in n(l_{\min}) \setminus \Xi_{l_{\min}}(k-1)} \lambda_{il_{\min}}(k-1) \\ &= F_{l_{\min}}(k-1) \quad (29) \\ \sum_{i \in n(l_{\min})} \lambda_{il_{\min}}(k) &= \sum_{i \in n(l_{\min}) \setminus \Xi_{l_{\min}}(k-1)} \lambda_{il_{\min}}(k) + \sum_{i \in \Xi_{l_{\min}}(k-1)} \lambda_{il_{\min}}(k) \\ &\quad \text{(since } \Xi_{l_{\min}}(k-1) \subseteq n(l_{\min}) \text{)} \\ &= F_{l_{\min}}(k-1) + \sum_{i \in \Xi_{l_{\min}}(k-1)} \max(\eta_{il_{\min}}(k), \mu_{il_{\min}}) \\ &\quad \text{(from (28) and (29))} \\ &= C_l \quad \text{(since } \Xi_{l_{\min}}(k-1) \neq \phi \text{)} \quad (30) \end{aligned}$$

Thus from (25) and (30)  $s \in S(k-1) \setminus S(k)$ ,  $\forall s \in m(i_{\min} l_{\min}) \cap S(k-1)$ . Thus  $S(k-1) \setminus S(k) \supseteq m(i_{\min} l_{\min}) \cap S(k-1)$ ,  $S(k) \subseteq S(k-1)$  by construction, and  $m(i_{\min} l_{\min}) \cap S(k-1) \neq \phi$ . Thus  $S(k)$  is a proper subset of  $S(k-1)$ .  $\square$

## C Proof of Lemma 2

The algorithm terminates in fewer number of iterations (at most  $(|\mathcal{L}, M)$  iterations) if the minimum rate requirements satisfy a particular property, i.e., all virtual sessions of the same session sharing a link at some point, have the same minimum rate requirements. To show this, as in the proof of Theorem 2, we need to consider link  $l_{\min}$ , the link which attains the minimum link control parameter amongst all those which carry at least one unsaturated virtual session. In the more general case, every unsaturated virtual session, of a particular session,  $i_{\min}$ , traversing through  $l_{\min}$  saturate. However in this

case, all unsaturated virtual sessions traversing  $l_{\min}$  saturate. Thus by  $|\mathcal{L}|$  iterations, all virtual sessions are saturated. The result follows. The formal proof is given below.

**Proof of Lemma 2:** Let  $S(k-1) \neq \phi$ . Consider link  $l_{\min}$ , where  $l_{\min}$  is as defined in proof of Theorem 2. Consider an unsaturated virtual session  $s$  traversing through  $l_{\min}$  (i.e.,  $\chi(s) \in n(l_{\min})$  and  $s \in S(k-1)$ ). Let  $r_s(k) < \eta_{\chi(s)l_{\min}}(k)$ . Since  $\mu_i = \mu_j$  if  $\chi(i) = \chi(j)$  and  $L_i \cap L_j \neq \phi$ ,  $\eta_{\chi(s)l}(k) = \max(\eta_l(k), \mu_s) \forall l \in L_s$ . Thus since  $r_s(k) < \eta_{\chi(s)l_{\min}}(k)$  and  $s \in S(k-1)$ ,  $\max(\eta_l(k), \mu_s) < \max(\eta_{l_{\min}}(k), \mu_s)$  for some  $l \in L_s$ . By definition  $\eta_{l_{\min}}(k) = \min_{l \in \cup_{p \in S(k-1)} L_p} \eta_l(k)$ . Thus  $\eta_{l_{\min}}(k) \leq \eta_l(k) \forall l \in L_s$ , since  $s \in S(k-1)$ . It follows from  $\max(\eta_l(k), \mu_s) < \max(\eta_{l_{\min}}(k), \mu_s)$  that  $\mu_s \geq \eta_{l_{\min}}(k)$ . Thus  $\eta_{\chi(s)l_{\min}}(k) = \mu_s \leq r_s(k)$ . The last inequality follows from the feasibility of  $\vec{r}(k)$ . This contradicts the assumption that  $r_s(k) < \eta_{\chi(s)l_{\min}}(k)$ . So  $r_s(k) \geq \eta_{\chi(s)l_{\min}}(k)$ . Now  $\lambda_{\chi(s)l_{\min}}(k) \leq \eta_{\chi(s)l_{\min}}(k)$  by Lemma 5. Thus  $r_s(k) \geq \lambda_{\chi(s)l_{\min}}(k)$ . Thus  $r_s(k) = \lambda_{\chi(s)l_{\min}}(k)$  for every unsaturated virtual session traversing through link  $l_{\min}$ .  $\sum_{i \in n(l_{\min})} \lambda_{il_{\min}}(k) = C_{l_{\min}}$  by (18). Thus every virtual session traversing through link  $l_{\min}$  and unsaturated at the end of iteration  $k-1$  becomes saturated at the end of iteration  $k$ . Thus  $\Xi_{l_{\min}}(k) = \phi$ . By choice, of  $l_{\min}$ ,  $\Xi_{l_{\min}}(k-1) \neq \phi$ . Let  $\Phi(k) = \{l : \Xi_l(k) \neq \phi\}$ .  $\Phi(k) \subseteq \Phi(k-1)$ , since  $\Xi_l(k) \subseteq \Xi_l(k-1)$ ,  $\forall l, k \geq 1$ . It follows that  $\Phi(k) \subset \Phi(k-1)$  and  $|\Phi(k)| < |\Phi(k-1)|$ ,  $\forall k \geq 1$ .  $|\Phi(0)| \leq |\mathcal{L}|$ . If  $\Phi(k) = \phi$ , then  $S(k) = \phi$  and  $\Phi(k)$  becomes empty in at most  $|\mathcal{L}|$  iterations. It follows that  $S(k) = \phi$  for  $k \geq |\mathcal{L}|$ . Thus the algorithm terminates in at most  $|\mathcal{L}|$  iterations. The lemma follows from Theorem 2.  $\square$

## D Asynchronous Distributed Algorithm

We first present a centralized algorithm for computation of maxmin fair rates and show that the receiver rates, forward session link rates and the source rates computed by the asynchronous distributed algorithm converge to the maxmin fair rates computed by the centralized algorithm. The centralized algorithm is similar to the synchronous distributed algorithm. The following are the differences.

1.  $\alpha(k) = \min_{l: \Xi_l(k-1) \neq \phi} \eta_l(k)$ ,  $\alpha_{il}(k) = \max(\alpha(k), \mu_{il})$
2. If  $s \in S(k-1)$ ,  $r_s(k) = \min_{l \in l_s} \alpha_{il}(k)$

3.  $S(k) = S(k-1) \setminus \{s : s \in S(k-1), \exists l \in L_s, \text{ s.t. } \lambda_{\chi(s)l}(k) = \alpha(k), \text{ and } \sum_{i \in n(l)} \lambda_{il}(k) = C_l\}$ .

Let  $\mathcal{L}(k) = \{l : \Xi_l(k-1) \neq \phi, \eta_l(k) = \alpha(k)\}$ . The following theorem shows that the rate allocations upon termination of this algorithm are max-min fair.

**Theorem 4** *The algorithm terminates in at most  $M$  iterations. Upon termination, the rate allocations are maxmin fair.*

**Proof of Theorem 4:** First we show that the algorithm terminates in  $M$  iterations. For this, it is sufficient to show that  $S(k) \subset S(k-1)$ , if  $S(k-1) \neq \phi$ . Consider any link  $l \in \mathcal{L}(k)$ . Let  $i_{\min} = \arg \min_{i \in \Xi_{l_{\min}}(k-1)} \mu_{il_{\min}}$ . Since  $\Xi_{l_{\min}} \neq \phi$ ,  $i_{\min}$  is well defined. Proceeding as in Theorem 2,  $\eta_l(k) \geq \mu_{i_{\min}l}$ . Thus  $\alpha(k) \geq \mu_{i_{\min}l}$  since  $\alpha(k) = \eta_l(k)$  as  $l \in \mathcal{L}(k)$ . Thus  $\lambda_{i_{\min}l}(k) = \alpha(k)$ . (It is easy to see that  $\lambda_{il_1}(k) = \alpha_{il_1}(k)$ ,  $\forall i \in \Xi_{l_1}(k-1)$ ,  $\forall l_1$ .) Since  $\eta_l(k) = \alpha(k)$ ,  $\lambda_{il}(k) = \alpha_{il}(k) = \eta_{il}(k)$ ,  $\forall i \in \Xi_l(k-1)$ . Also  $\lambda_{il}(k) = \lambda_{il}(k-1)$  if  $i \in n(l) \setminus \Xi_l(k-1)$ . Thus  $\sum_{i \in n(l)} \lambda_{il}(k) = F_l(k-1) + \sum_{i \in \Xi_l(k-1)} \eta_{il}(k) = C_l$ . Thus every virtual session in  $m(i_{\min}, l) \cap S(k-1)$  is saturated. Since  $i \in \Xi_l(k-1)$ , there is at least one such virtual session. Thus  $S(k) \subset S(k-1)$ .

Proceeding as in Lemma 3, it can be shown that

$$\eta_l(k) \geq \eta_l(k-1), \forall l, k, k \geq 1. \quad (31)$$

$\alpha(1) \geq 0 = \alpha(0)$ . Let  $k > 1$ . Now  $\alpha(k) = \min_{l: \Xi_l(k-1) \neq \phi} \eta_l(k)$  and  $\min_{l: \Xi_l(k-1) \neq \phi} \eta_l(k) \geq \min_{l: \Xi_l(k-2) \neq \phi} \eta_l(k-1)$ , since  $\{l : \Xi_l(k-1) \neq \phi\} \subseteq \{l : \Xi_l(k-2) \neq \phi\}$  (as  $\Xi_l(k-1) \subseteq \Xi_l(k-2)$ ) and  $\eta_l(k) \geq \eta_l(k-1) \forall l$ .

$$\text{Thus } \alpha(k) \geq \alpha(k-1) \forall k. \quad (32)$$

It follows that  $\lambda_{il}(k) \geq \lambda_{il}(k-1) \forall i, l$  and  $r_s(k) \geq r_s(k-1) \forall s$  and

$$\lambda_{il}(k) \geq \lambda_{il}(k-1) \forall i, l \quad (33)$$

Thus  $r_s(k) \geq r_s(0) = \mu_s$ . Also  $\lambda_{il}(k) = \alpha_{il}(k) \leq \eta_{il}(k)$ ,  $\forall i \in \Xi_l(k-1)$  and  $l$ . Thus the rate vectors satisfy the capacity condition at all links at all iterations.

$$\text{So } \vec{r}(k) \text{ is a feasible rate vector for all } k. \quad (34)$$

Let  $s \in S(k-1) \setminus S(k)$ . This means there exists a link  $l \in L_s$  s.t.  $\lambda_{\chi(s)l}(k) = \alpha(k)$  and  $\sum_{i \in n(l)} \lambda_{il}(k) = C_l$ . Clearly,  $r_s(k) = \alpha(k)$  in this case. Consider

any subsequent iteration  $t$ ,  $t \geq k$ . Since  $\lambda_{il}(t) \geq \lambda_{il}(k) \forall i \in n(l)$ , and  $\vec{r}(t)$  satisfies the capacity condition at link  $l$ ,  $\lambda_{il}(t) = \lambda_{il}(k) \forall i \in n(l)$ . Since  $s \in S(k-1) \setminus S(k)$ ,  $r_s(t) = r_s(k)$ . Thus  $r_s(t) = \lambda_{\chi(s)l}(t)$ . If  $\lambda_{il}(t) > \lambda_{\chi(s)l}(t)$ , then  $\lambda_{il}(k) > \lambda_{\chi(s)l}(k) = \alpha(k)$ . Thus  $\lambda_{il}(k) = \mu_{il}$  and hence  $\lambda_{il}(t) = \mu_{il}$ . It follows that if  $r_j(t) > r_s(t)$ , for some  $j$  s.t.  $l \in L_j$ , then  $r_j(t) \leq \mu_{\chi(j)l}$ . Since  $\lambda_{il}(t) = \lambda_{il}(k) \forall i \in n(l)$  and  $\sum_{i \in n(l)} \lambda_{il}(k) = C_l$ ,  $\sum_{i \in n(l)} \lambda_{il}(t) = C_l$ . Thus link  $l$  is bottlenecked w.r.t. virtual session  $s$  for all iterations  $t \geq k$ . Since every virtual session is saturated when the algorithm terminates, every virtual session has a bottleneck link upon termination and thus the rate vector obtained upon termination is maxmin fair by Lemma 1.  $\square$

**Lemma 8** *If  $s \in S(k-1) \setminus S(k)$ , then there exists a link  $l$  s.t.  $l \in L_s \cap \mathcal{L}(k)$ ,  $\lambda_{\chi(s)l}(k) = \alpha(k)$  and  $\chi(s) \in \Xi_l(k-1) \setminus \Xi_l(k)$ . Call this link “saturation-link”.*

**Proof of Lemma 8:** Since  $s \in S(k-1) \setminus S(k)$ , there exists  $l \in L_s$ , such that  $\lambda_{\chi(s)l}(k) = \alpha(k)$  and  $\sum_{i \in n(l)} \lambda_{il}(k) = C_l$ . For all  $j \in m(\chi(s), l) \cap S(k-1)$ ,  $\lambda_{\chi(j)l}(k) = \alpha(k)$  and  $\sum_{i \in n(l)} \lambda_{il}(k) = C_l$ . Thus  $j \notin S(k)$ . Thus  $\chi(s) \notin \Xi_l(k)$ . However,  $\chi(s) \in \Xi_l(k-1)$ . So  $\chi(s) \in \Xi_l(k-1) \setminus \Xi_l(k)$ . Let  $\alpha(k) < \eta_l(k)$ .  $\lambda_{\chi(s)l}(k) < \eta_l(k) \leq \eta_{\chi(s)l}(k)$ . Clearly,  $\lambda_{il}(k) = \alpha_{il}(k) \leq \eta_{il}(k)$ ,  $\forall i \in n(l)$ . Thus  $F_l(k-1) + \sum_{i \in \Xi_l(k-1)} \lambda_{il}(k) < F_l(k-1) + \sum_{i \in \Xi_l(k-1)} \eta_{il}(k) = C_l$ . This is a contradiction. Thus  $\alpha(k) = \eta_l(k)$ . Since  $l \in L_s$ , and  $s \in S(k-1)$ ,  $\Xi_l(k-1) \neq \phi$ . Thus  $l \in \mathcal{L}(k)$ .  $\square$

**Lemma 9** *Let  $l \in \mathcal{L}(k)$ . If  $i \in \Xi_l(t)$ , then  $\eta_{ii}(t+1) = \eta_{il}(k)$ ,  $\forall t \geq k$ .*

**Proof of Lemma 9:** Since  $l \in \mathcal{L}(k)$ ,  $\Xi_l(k-1) \neq \phi$ . Now,  $\forall i \in \Xi_l(k-1)$ ,  $\lambda_{il}(k) = \alpha_{il}(k)$  and  $\alpha_{il}(k) = \eta_{il}(k)$ , since  $l \in \mathcal{L}(k)$ . Thus  $\sum_{i \in n(l)} \lambda_{il}(k) = F_l(k-1) + \sum_{i \in \Xi_l(k-1)} \eta_{il}(k) = C_l$ . This means if  $i \in \Xi_l(k)$ ,  $\lambda_{il}(k) > \alpha(k)$ . The latter can happen only if  $\alpha(k) < \mu_{il}$ . It follows that  $\eta_l(k) < \mu_{il}$ . Thus  $\eta_{ii}(k) = \mu_{il}$  if  $i \in \Xi_l(k)$ . If  $i \in \Xi_l(k)$ ,  $\lambda_{il}(k) \leq \eta_{il}(k) = \mu_{il}$ . From feasibility of  $\vec{r}(k)$  ((34)),  $\lambda_{il}(k) = \mu_{il}$ . It follows that  $\lambda_{il}(k) = \eta_{il}(k)$ , if  $i \in \Xi_l(k)$ . If  $i \in \Xi_l(k-1) \setminus \Xi_l(k)$ ,  $\lambda_{il}(k) = \alpha_{il}(k) = \eta_{il}(k)$ .

$$\text{Thus } \lambda_{il}(k) = \eta_{il}(k), \text{ if } i \in \Xi_l(k-1). \quad (35)$$

Consider any time  $t+1$ ,  $t \geq k$ .

$$\text{If } i \in \Xi_l(k-1) \setminus \Xi_l(t), \lambda_{il}(t) \geq \lambda_{il}(k) = \eta_{il}(k). \quad (36)$$

The first inequality follows from (33) and the second equality from (35). Let  $\eta_{il}(t+1) \neq \eta_{il}(k)$ , for some  $i \in \Xi_l(t)$ . Thus  $\Xi_l(t) \neq \phi$ . Also, from (31), this means that  $\eta_{il}(t+1) > \eta_{il}(k)$ .

$$\begin{aligned}
 & \sum_{i \in \Xi_l(t)} \eta_{il}(t+1) + F_l(t) \\
 > & \sum_{i \in \Xi_l(t)} \eta_{il}(k) + F_l(t) \\
 = & \sum_{i \in \Xi_l(t)} \eta_{il}(k) + \sum_{i \in \Xi_l(k-1) \setminus \Xi_l(t)} \lambda_{il}(t) + F_l(k-1) \\
 \geq & \sum_{i \in \Xi_l(t)} \eta_{il}(k) + \sum_{i \in \Xi_l(k-1) \setminus \Xi_l(t)} \eta_{il}(k) + F_l(k-1) \text{ (from (36))} \\
 = & \sum_{i \in \Xi_l(k-1)} \eta_{il}(k) + F_l(k-1) \\
 = & C_l
 \end{aligned}$$

This contradicts the definition of  $\eta_l(t+1)$ . It follows that  $\forall i \in \Xi_l(t)$ ,  $\eta_{il}(t+1) = \eta_{il}(k)$ .  $\square$

**Lemma 10** *If  $i \in \Xi_l(p) \setminus \Xi_l(p+1)$ , then  $\alpha(p+1) \geq \mu_{il}$ .*

**Proof of Lemma 10:** If  $s \in m(i, l) \cap S(p)$ , then  $s \in S(p) \setminus S(p+1)$ . Let  $l_1$  be the saturation link of virtual session  $s$ . By Lemma 8,  $\lambda_{il_1}(p+1) = \alpha(p+1)$ , and  $\lambda_{il_1}(p+1) \geq \mu_s$  by feasibility of  $\vec{r}(p+1)$ ((34)). Thus  $\alpha(p+1) \geq \mu_s$ ,  $\forall s \in m(i, l) \cap S(p)$ . If  $s \in m(i, l) \setminus S(p)$ , then let  $s \in S(k) \setminus S(k+1)$ ,  $k < p$ . Clearly,  $r_s(k+1) \leq \lambda_{\chi(s)l_2}(k+1) = \alpha(k+1)$ , for some  $l_2 \in L_s$ . Since  $k+1 < p+1$ , by (32),  $r_s(k+1) \leq \alpha(p+1)$  and  $r_s(k+1) = r_s(p+1)$ , since  $s \notin S(k+1)$  and  $k+1 < p+1$ . Thus  $r_s(p+1) \leq \alpha(p+1)$  and  $r_s(p+1) \geq \mu_s$  from feasibility of  $\vec{r}(p+1)$ ((34)). Thus  $\alpha(p+1) \geq \mu_s \forall s \in m(i, l) \setminus S(p)$ . It follows that  $\alpha(p+1) \geq \mu_s, \forall s \in m(i, l)$ . Thus  $\alpha(p+1) \geq \mu_{il}$ .

**Lemma 11** *If  $s \in S(k) \setminus S(k+1)$ , then  $r_s(p) = \alpha(k+1), \forall p > k$ . If  $i \in \Xi_l(k) \setminus \Xi_l(k+1)$ ,  $\lambda_{il}(p) = \alpha(k+1), \forall p > k$ .*

**Proof of Lemma 11:** Clearly  $r_s(p) = r_s(k+1)$ . Since  $s \in S(k) \setminus S(k+1)$ , there exists a link  $l$  s.t.  $\lambda_{\chi(s)l}(k+1) = \alpha(k+1)$ . Thus  $r_s(k+1) \leq \alpha(k+1)$ . Also  $r_s(k+1) \geq \alpha(k+1)$ , since  $r_s(k+1) = \min_{l \in L_s} \alpha_{il}(k+1)$ . Thus  $r_s(k+1) = \alpha(k+1)$ . The first part follows.

If  $i \in \Xi_l(k) \setminus \Xi_l(k+1)$ , then  $r_s(p) = \alpha(k+1) \forall s \in m(i, l) \cap S(k)$ . If  $s \in m(i, l) \setminus S(k)$ , then  $s \in S(t) \setminus S(t+1)$ , for some  $t < k$ . Thus  $r_s(p) = \alpha(t+1) \leq \alpha(k+1)$  ((32)). The result follows since  $m(i, l) \cap S(k) = \phi$ .  $\square$

Let the system configuration stop changing after  $t = t_0$ . Thus  $n(l) = \kappa_l$ , for  $t \geq t_0$ . Let  $D$  be the maximum source-destination roundtrip time of a session. Any session  $i$  feedback rate packet sent in response to a forward rate packet sent after  $t_0$  reaches the origin of link  $l$  with minimum rate value equal to  $\mu_{il}$  and rate value not less than  $\mu_{il}$ . Thus any such feedback rate packet sets  $\hat{\mu}_{il}$  equal to  $\mu_{il}$ . Only session  $i$  feedback rate packets alter  $b_{il}$  and  $\hat{\mu}_{il}$ . Thus  $b_{il} \geq \mu_{il}$  and  $\hat{\mu}_{il} = \mu_{il}$  always after the first such feedback rate packet reaches the origin of link  $l$ . Also  $\psi_{il} \geq \mu_{il}$  after the first such feedback rate packet reaches the origin of link  $l$ . Any feedback rate packet sent in response to the forward rate packet sent after  $t_0$  reaches the source with rate value not less than the minimum rate of any session receiver. Thus source rate is always greater than or equal to the maximum of the minimum receiver rates after the first such feedback rate packet reaches the source. This happens before  $t_0 + 2D$ . The second and subsequent forward rate packets sent after  $t_0$  set  $f_{il} \geq \mu_{il}$ . This is because such forward rate packets are sent from the source with rate value not less than the minimum rate of any session  $i$  receiver. Also any such forward rate packet finds  $b_{il_1} \geq \mu_{il_1}$  and  $\psi_{il_1} \geq \mu_{il_1}$  for all links  $l_1$  on its path and  $\hat{\mu}_{il_1} = \mu_{il_1} \geq \mu_{il}$  if  $l_1$  lies on session  $i$  path between session  $i$  source and the origin of link  $l$ . Thus any such forward rate packet reaches the origin of link  $l$  with rate value not less than  $\mu_{il}$ . Since  $\psi_{il} \geq \mu_{il}$  and  $b_{il} \geq \mu_{il}$ , the forward rate packet sets  $f_{il}$  to a value not less than  $\mu_{il}$ . Since  $f_{il}$  alters only upon arrival of a session  $i$  forward rate packet,  $f_{il} \geq \mu_{il}$  always after the arrival of the second forward rate packet sent after  $t_0$ . Such a forward rate packet arrives at the origin of link  $l$  before  $t_0 + 2.5D$ . So  $\rho_i \geq \max_{s: \chi(s)=i} \mu_s$ ,  $f_{il} \geq \mu_{il}$ ,  $b_{il} \geq \mu_{il}$  and  $\hat{\mu}_{il} = \mu_{il}$ ,  $\forall i, \forall l$ , s.t.  $i \in n(l)$ , and for all  $t \geq T_0$ , where  $T_0 = t_0 + 2.5D$ .

**Lemma 12** *Consider a link  $l$  just after a link control parameter estimation for  $t \geq T_0$ . If at least one session is unmarked, then  $\sum_{i:w_{il}=0} \mu_{il} \leq C_l - \sum_{i:w_{il}=1} f_{il}$ . Also, if at least one session is unmarked,  $\sum_{i:w_{il}=0} \mu_{il} = C_l - \sum_{i:w_{il}=1} f_{il}$  implies that no session is marked, i.e.,  $w_{il} = 0$  for all  $i \in n(l)$ .*

**Proof of Lemma 12:** If the estimation subroutine terminates at the first “end subroutine” statement, then  $n(l) = \phi$  and the lemma holds by vacuity.

If the subroutine terminates at the second “end subroutine” statement, then all sessions are marked upon termination, and the lemma holds by vacuity. If it terminates at the third or the fourth “end subroutine” statement, then all sessions are unmarked and the lemma holds trivially. So let the subroutine terminate at the last such statement. Thus the condition that  $\sum_{i:w_{il}=0} \mu_{il} = C_l - \sum_{i:w_{il}=1} f_{il}$  is tested. Observe that the value of  $f_{il}$  is not changed in the subroutine, but the value of  $w_{il}$  changes. However once  $w_{il} = 0$ , the value of  $w_{il}$  does not change any further. Let  $w_{il}^1$  indicate the value of these variables, just before the above condition is tested. Let  $w_{il}^2$  denote the values just before the first estimation of the link control parameter after this test. Let  $\sum_{i:w_{il}^1=0} \mu_{il} = C_l - \sum_{i:w_{il}^1=1} f_{il}$ . Then  $\{i : w_{il}^2 = 1\} \subset \{i : w_{il}^1 = 1\}$ . This is because there exists at least one  $i$  s.t.  $w_{il}^1 = 1$ , and  $f_{il} = \beta_l$  as  $\beta_l$  exists (otherwise termination takes place at the fourth “end subroutine” statement). For this  $i$ ,  $w_{il}^2 = 0$ . As mentioned before,  $w_{il}^1 \neq 1$  means that  $w_{il}^2 \neq 1$ . Also  $\{i : w_{il}^1 = 0\} \subset \{i : w_{il}^2 = 0\}$ .

$$\begin{aligned}
 \sum_{i:w_{il}^2=0} \mu_{il} &= \sum_{i:w_{il}^1=0} \mu_{il} + \sum_{i:w_{il}^1=1, w_{il}^2=0} \mu_{il} \\
 &\quad (\text{since } \{i : w_{il}^1 = 0\} \subset \{i : w_{il}^2 = 0\}) \\
 &= C_l - \sum_{i:w_{il}^1=1} f_{il} + \sum_{i:w_{il}^1=1, w_{il}^2=0} \mu_{il} \\
 &= C_l - \sum_{i:w_{il}^2=1} f_{il} - \sum_{i:w_{il}^1=1, w_{il}^2=0} f_{il} + \sum_{i:w_{il}^1=1, w_{il}^2=0} \mu_{il} \\
 &\quad (\text{since } \{i : w_{il}^2 = 1\} \subset \{i : w_{il}^1 = 1\}) \\
 &< C_l - \sum_{i:w_{il}^2=1} f_{il} \tag{37}
 \end{aligned}$$

The last inequality follows because  $f_{il} \geq \mu_{il}$  for  $t \geq T_0$ ,  $\forall i \in n(l)$  and there exists a  $i$  s.t.  $w_{il}^1 = 1$ ,  $f_{il} = \beta_l$  and  $f_{il} > \mu_{il}$ . Clearly  $w_{il}^2 = 0$  for such  $i$ . Thus if  $\sum_{i:w_{il}^1=0} \mu_{il} = C_l - \sum_{i:w_{il}^1=1} f_{il}$  then  $\sum_{i:w_{il}^2=0} \mu_{il} < C_l - \sum_{i:w_{il}^2=1} f_{il}$ . If  $\sum_{i:w_{il}^1=0} \mu_{il} \neq C_l - \sum_{i:w_{il}^1=1} f_{il}$ , then  $w_{il}^1 = w_{il}^2$ ,  $\forall i \in n(l)$ . Also, then  $\sum_{i:w_{il}^1=0} \mu_{il} < C_l - \sum_{i:w_{il}^1=1} f_{il}$  (otherwise termination takes place at the third “end subroutine” statement). Thus  $\sum_{i:w_{il}^2=0} \mu_{il} < C_l - \sum_{i:w_{il}^2=1} f_{il}$ . So (37) holds in both these cases.

Let  $w_{il}^3$  be the session link saturation values when the subroutine terminates. If there is only one estimation of the link control parameter, then

$w_{il}^2 = w_{il}^3$  for all  $i \in n(l)$ . From (37), the lemma clearly holds in this case. Let a second estimation take place. It follows that  $\{i : w_{il}^3 = 1\} \subset \{i : w_{il}^2 = 1\}$  and  $\{i : w_{il}^2 = 0\} \subset \{i : w_{il}^3 = 0\}$ . Thus

$$\begin{aligned}
\sum_{i:w_{il}^3=0} \mu_{il} &= \sum_{i:w_{il}^2=0} \mu_{il} + \sum_{i:w_{il}^2=1, w_{il}^3=0} \mu_{il} \\
&\quad (\text{since } \{i : w_{il}^1 = 0\} \subset \{i : w_{il}^2 = 0\}) \\
&< C_l - \sum_{i:w_{il}^2=1} f_{il} + \sum_{i:w_{il}^2=1, w_{il}^3=0} \mu_{il} \quad (\text{from (37)}) \\
&= C_l - \sum_{i:w_{il}^3=1} f_{il} - \sum_{i:w_{il}^2=1, w_{il}^3=0} f_{il} + \sum_{i:w_{il}^2=1, w_{il}^3=0} \mu_{il} \\
&\quad (\text{since } \{i : w_{il}^3 = 1\} \subset \{i : w_{il}^2 = 1\}) \\
&\leq C_l - \sum_{i:w_{il}^3=1} f_{il} \quad (\text{since } f_{il} \geq \mu_{il} \forall i \in n(l), \forall t \geq T_0)
\end{aligned}$$

The subroutine terminates with markings determined by  $w_{il}^3$ s. Thus the lemma holds in this case.  $\square$

Henceforth  $w_{il}^o, f_{il}^o$  indicate the values at the end of the **estimate-link-control-parameter** which computes the current  $\psi_l$ . A session  $i$  is marked on a link  $l$  on its path if  $w_{il} = 1$ . Also  $\mathcal{M}_l$  is the set of sessions marked after the last estimation of link control parameter, i.e.,  $\mathcal{M}_l = \{i : i \in n(l), w_{il}^o = 1\}$ .

**Lemma 13** *If  $\mathcal{M}_l \subset n(l)$ , then  $\sum_{i \in n(l) \setminus \mathcal{M}_l} \psi_{il} = C_l - \sum_{i \in \mathcal{M}_l} f_{il}^o$ . for all  $t$ . If  $\mathcal{M}_l = n(l) \neq \phi$ , then  $\psi_l = C_l - \sum_{i \in n(l)} f_{il}^o + \max_{i \in n(l)} f_{il}^o$  for all  $t$ .*

**Proof of Lemma 13:** Since  $\mathcal{M}_l \subset n(l)$ , the subroutine which computes the current  $\psi_l$  does not terminate at the first or the second “end-subroutine” statement. If the subroutine terminates at the third “end-subroutine” statement, then  $w_{il}^o = 0, \forall i \in n(l)$ , i.e.,  $\mathcal{M}_l = \phi$ , and  $\sum_{i \in n(l)} \psi_{il} = C_l$ . Thus the lemma holds in this case. If the subroutine terminates at the fourth “end-subroutine” statement, then  $\psi_{il} = \hat{\mu}_{il} \forall i \in n(l)$ . Also, in this case,  $\sum_{i \in n(l)} \hat{\mu}_{il} = C_l$  and  $\mathcal{M}_l = \phi$ . Thus the lemma holds. The lemma clearly holds if the subroutine terminates at the last “end-subroutine” statement.

If  $\mathcal{M}_l = n(l) \neq \phi$ , then the subroutine terminates at the second “end-subroutine”. All sessions are marked before and after the estimation. Hence the lemma holds by the operation of the algorithm.  $\square$



**Lemma 14 (Marking-Consistency)** *If  $i \in \mathcal{M}_l$ , then  $f_{il}^o < \psi_l$ , for all  $t \geq T_0$ .*

**Proof of Lemma 14:** If the routine terminates at the first, third and fourth “end subroutine” statements, then no session is marked at link  $l$ , and the lemma holds by vacuity. If the subroutine terminates at the second “end subroutine” statement, then the lemma holds by the operation of the subroutine. Let the subroutine terminate at the last “end subroutine” statement. Again the lemma holds by the operation of the subroutine, if there is only one estimate after the minimum rate test. Now, let there be two estimations of  $\psi_l$  after the minimum rate test.

Let  $\mathcal{A}_l$  be the set of sessions marked before the first computation of  $\psi_l$ , after the minimum rate test. Let  $\psi_l^1$  be the first estimate of the link control parameter, after the minimum rate test. Thus  $\sum_{i \in n(l) \setminus \mathcal{A}_l} \psi_{il}^1 = C_l - \sum_{i \in \mathcal{A}_l} f_{il}^o$ . Let  $\mathcal{Z}_l$  be the set of sessions unmarked after computing  $\psi_l^1$ . While proving Lemma 12, we have proved that if the subroutine terminates at the last “end subroutine” statement,

$$C_l - \sum_{i \in \mathcal{A}_l} f_{il}^o > \sum_{i \in n(l) \setminus \mathcal{A}_l} \mu_{il} \quad (38)$$

Since the subroutine makes a second estimation of link control parameter,  $\mathcal{Z}_l \neq \phi$ . The last estimation is either made after  $T_0$  or is the last one before  $T_0$  and hence  $f_{il}^o \geq \mu_{il} \forall i \in n(l)$ . Let  $\psi_l^2$  be the second estimate of the link control parameter.

$$\begin{aligned} \sum_{i \in n(l) \setminus (\mathcal{A}_l \setminus \mathcal{Z}_l)} \psi_{il}^2 &= C_l - \sum_{i \in \mathcal{A}_l \setminus \mathcal{Z}_l} f_{il}^o \\ &= C_l - \sum_{i \in \mathcal{A}_l} f_{il}^o + \sum_{i \in \mathcal{Z}_l} f_{il}^o \\ &> \sum_{i \in n(l) \setminus \mathcal{A}_l} \mu_{il} + \sum_{i \in \mathcal{Z}_l} \mu_{il} \\ &\quad (\text{from (38) and since } f_{il}^o \geq \mu_{il} \forall i \in n(l)) \\ &= \sum_{i \in n(l) \setminus (\mathcal{A}_l \setminus \mathcal{Z}_l)} \mu_{il} \end{aligned} \quad (39)$$

$$\text{Also } \sum_{i \in n(l) \setminus (\mathcal{A}_l \setminus \mathcal{Z}_l)} \psi_{il}^2 = C_l - \sum_{i \in (\mathcal{A}_l \setminus \mathcal{Z}_l)} f_{il}$$

$$\begin{aligned}
&= \left( C_l - \sum_{i \in \mathcal{A}_l} f_{il} \right) + \sum_{i \in \mathcal{Z}_l} f_{il} \\
&= \sum_{i \in n(l) \setminus \mathcal{A}_l} \psi_{il}^1 + \sum_{i \in \mathcal{Z}_l} f_{il} \\
&\geq \sum_{i \in n(l) \setminus (\mathcal{A}_l \setminus \mathcal{Z}_l)} \psi_{il}^1 \quad (\text{if } i \in \mathcal{Z}_l, f_{il} \geq \psi_{il}^1) \quad (40)
\end{aligned}$$

From (40),  $\psi_{il}^2 \geq \psi_{il}^1, \forall i \in n(l) \setminus (\mathcal{A}_l \setminus \mathcal{Z}_l)$ . From (39), and the fact that  $\psi_{il}^2 \geq \hat{\mu}_{il} = \mu_{il} \forall i, l$  s.t.  $i \in n(l)$  and  $t \geq T_0$ ,  $\psi_{il}^2 > \mu_{il}$  for some  $i \in n(l) \setminus (\mathcal{A}_l \setminus \mathcal{Z}_l)$ . Thus  $\psi_{il}^2 = \psi_l^2$  for such an  $i$ . Thus  $\psi_l^2 = \psi_{il}^2 \geq \psi_{il}^1 \geq \psi_l^1$ . If  $i \in (\mathcal{A}_l \setminus \mathcal{Z}_l)$ , then  $f_{il}^o < \psi_l^1$ . It follows that  $f_{il}^o < \psi_l^2$ , if  $i \in (\mathcal{A}_l \setminus \mathcal{Z}_l)$ . Note that  $\mathcal{M}_l = \mathcal{A}_l \setminus \mathcal{Z}_l$ . Thus the result follows.  $\square$

Let  $t^-$  be the time of the last computation before  $t$ .

**Lemma 15** *There exists a time  $T_1 \geq T_0$  such that for all  $t \geq T_1^-$ ,*

1. *For all links  $l$ , and for all  $i \in \Xi_l(0)$ ,  $\psi_{il} \geq \eta_{il}(1)$ .*
2.  *$b_{il} \geq \alpha_{il}(1), \forall l, i \in \Xi_l(0)$ .*
3.  *$\rho_i \geq \max(\alpha(1), \max_{s: \chi(s)=i} \mu_s), \forall i$  s.t.  $i \in \Xi_l(0)$  for some link  $l$ .*
4.  *$f_{il} \geq \alpha_{il}(1), \forall l, i \in \Xi_l(0)$ .*
5. *If  $l \in \mathcal{L}(1)$ ,  $\mathcal{M}_l \subseteq n(l) \setminus \Xi_l(0)$ .*
6. *If  $l \in \mathcal{L}(1)$ , then  $\psi_{il} = \alpha_{il}(1), \forall i \in \Xi_l(0)$ . If  $i \in \Xi_l(0) \setminus \Xi_l(1)$ , and  $l \in \mathcal{L}(1)$ ,  $\psi_{il} = \alpha(1)$ .*
7. *If  $l \in \mathcal{L}(1)$ , then  $f_{il} = \alpha_{il}(1)$  for all  $i \in \Xi_l(0)$ . If  $i \in \Xi_l(0) \setminus \Xi_l(1)$ , and  $l \in \mathcal{L}(1)$ ,  $f_{il} = \alpha(1)$ .*
8. *If  $i \in \Xi_l(0) \setminus \Xi_l(1)$ ,  $f_{il} = \alpha(1)$ .*
9. *If  $i \in n(l) \setminus \Xi_l(1)$ , and  $l \notin \mathcal{L}(1)$ ,  $w_{il} = 1$ .*

*If virtual session  $s \in S(0) \setminus S(1)$ , then  $\sigma_s = \alpha(1)$ , for all  $t \geq T_1$ .*

**Proof of Lemma 15:** First we prove (1). Consider a link  $l$ .  $\Xi_l(0) = n(l)$ . Let  $t \geq T_0$ . First let  $\mathcal{M}_l \subset n(l)$ . Let  $\psi_l < \eta_l(1)$ . If  $i \in \mathcal{M}_l$ , by Lemma 14,  $f_{il}^o < \psi_l < \eta_l(1) \leq \eta_{il}(1)$ . Also let  $\mathcal{M}_l \neq \phi$ .

$$\begin{aligned}
\sum_{i \in n(l) \setminus \mathcal{M}_l} \psi_{il} &= C_l - \sum_{i \in \mathcal{M}_l} f_{il}^o \text{ (Lemma 13)} \\
&> C_l - \sum_{i \in \mathcal{M}_l} \eta_{il}(1) \\
&\quad \text{(since } \mathcal{M}_l \neq \phi, \text{ and } f_{il} < \eta_{il}(1), \text{ if } i \in \mathcal{M}_l) \\
&= \sum_{i \in n(l) \setminus \mathcal{M}_l} \eta_{il}(1) \tag{41}
\end{aligned}$$

Since  $\hat{\mu}_{il} = \mu_{il}$  for all  $t \geq T_0$ , this means that  $\psi_l > \eta_l(1)$  and hence  $\psi_{il} \geq \eta_{il}(1) \forall i \in n(l)$ . Now let  $\mathcal{M}_l = \phi$ .

$$\begin{aligned}
\sum_{i \in n(l)} \psi_{il} &= C_l \text{ (Lemma 13)} \\
\sum_{i \in n(l)} \eta_{il}(1) &= C_l
\end{aligned}$$

If  $\psi_{il} > \eta_{il}(1)$  for some  $i \in n(l)$ , then  $\psi_{il} = \psi_l$ , since  $\hat{\mu}_{il} = \mu_{il}$  for all  $t \geq T_0$ . Hence  $\psi_l > \eta_{il}(1) \geq \eta_l(1)$ . Again since  $\hat{\mu}_{il} = \mu_{il}$ , for all  $t \geq T_0$ , this means that  $\psi_{jl} \geq \eta_{jl}(1)$  for all  $j \in n(l)$ . This means  $\sum_{i \in n(l)} \psi_{il} > \sum_{i \in n(l)} \eta_{il}(1)$ . However this is not true. Thus  $\psi_{il} \leq \eta_{il}(1)$  for all  $i \in n(l)$ . Arguing similarly  $\eta_{il}(1) \leq \psi_{il} \forall i \in n(l)$ . So  $\psi_{il} = \eta_{il}(1) \forall i \in n(l)$ .

Now let all sessions traversing  $l$  be marked, i.e.,  $\mathcal{M}_l = n(l)$ . If  $\max_{i \in n(l)} f_{il}^o \geq \eta_l(1)$ , then since by Lemma 14,  $\psi_l > \max_{i \in n(l)} f_{il}^o$ ,  $\psi_l > \eta_l(1)$ . Now let  $\max_{i \in n(l)} f_{il}^o < \eta_l(1)$ . It follows that  $f_{il}^o < \eta_l(1)$ , for all sessions  $i \in n(l)$ . Let  $j$  be a session in  $n(l)$  for which  $f_{jl}^o = \max_{i \in n(l)} f_{il}^o$ .

$$\begin{aligned}
\psi_l &= C_l - \sum_{i \in n(l)} f_{il}^o + \max_{i \in n(l)} f_{il}^o \\
&= C_l - \sum_{i \in n(l), i \neq j} f_{il}^o \\
&> C_l - (|n(l)| - 1) \eta_l(1) \\
&= \sum_{i \in n(l)} \eta_{il}(1) - (|n(l)| - 1) \eta_l(1)
\end{aligned}$$

$$\begin{aligned}
&\geq \sum_{i \in n(l)} \eta_l(1) - (|n(l)| - 1) \eta_l(1) \\
&\quad \text{(since } \eta_{il}(1) \geq \eta_l(1) \forall i \in n(l)\text{)} \\
&= \eta_l(1)
\end{aligned}$$

Thus (1) holds for all  $t \geq T_0$ .

Now we prove (2). Let  $i \in \Xi_l(0)$ . Let  $\alpha(1) \leq \mu_{il}$ . Thus  $\alpha_{il}(1) = \mu_{il}$  since  $\hat{\mu}_{il} = \mu_{il}$  as  $t \geq T_0$ . Since  $b_{il} \geq \mu_{il}$ , for all  $t \geq T_0$ ,  $b_{il} \geq \alpha_{il}(1)$ ,  $\forall t \geq T_0$ . Now let  $\alpha(1) > \mu_{il}$ . Consider a session  $i$  forward rate packet leaving the origin of link  $l$  after  $T_0$ . It follows that  $\alpha(1) > \mu_{il_1}$  for all links  $l_1$  on session  $i$  path downstream of the origin of  $l$  (between the origin and the receivers). From (1),  $\psi_{il_1} \geq \eta_{il_1}(1) \geq \alpha_{il_1}(1) > \mu_{il_1}$ . Since  $\hat{\mu}_{il_1} = \mu_{il_1}$  as  $t \geq T_0$ ,  $\psi_{il_1} = \psi_{l_1}$ . Thus  $\psi_{l_1} = \psi_{il_1} \geq \alpha_{il_1}(1) = \alpha(1)$ . If it leaves with a rate strictly less than  $\alpha(1)$ , since  $\psi_{l_1} \geq \alpha(1)$ , for all downstream links  $l_1$  (including  $l$ ) and  $t \geq T_0$ ,  $u_p = 0$  in the corresponding feedback rate packet reaching the origin of link  $l$ . This sets  $b_{il} = \infty$ . If the rate packet leaves with a rate value greater than or equal to  $\alpha(1)$ , consider the path of session  $i$  from the origin of link  $l$  to one of the downstream receivers of session  $i$ . If the rate value of the forward rate packet all along this path is greater than or equal to  $\alpha(1)$ , clearly the feedback rate packet reaches the origin of link  $l$ , through link  $l$  with a rate value greater than or equal to  $\alpha(1)$ . Thus this feedback rate packet sets  $b_{il} \geq \alpha(1)$ . If the rate value of the forward rate packet all along this path is not greater than or equal to  $\alpha(1)$ , then consider the node  $n$  on its path closest to the origin of link  $l$ , such that a forward rate packet leaves the node with a rate value strictly less than  $\alpha(1)$ . Thus the session  $i$  forward rate packet reaches node  $n$  with rate value greater than or equal to  $\alpha(1)$ . Repeating one of the previous arguments, a feedback rate packet returns to this node with  $u_p = 0$ . Consequently, a feedback rate packet leaves node  $n$  with rate value not less than that of the forward rate packet which reaches node  $n$ . As we argued before, this is greater than or equal to  $\alpha(1)$ . Rate values in feedback rate packets do not decrease as they move towards the source. So a feedback rate packet reaches the origin of link  $l$  with rate value greater than or equal to  $\alpha(1)$ . This sets  $b_{il}$  to a value not less than  $\alpha(1)$ . The argument ensures that a session  $i$  feedback rate packet returning in response to a forward rate packet sent after  $T_0$  sets  $b_{il}$  to a value not less than  $\alpha(1)$ . Also the value of  $b_{il}$  changes only when session  $i$  feedback rate packets reach the origin of link  $l$ . A feedback rate packet sent in response to a forward rate packet sent

after  $T_0$  reaches the origin of link  $l$  in at most  $2D$ <sup>§</sup> time units after  $T_0$ . Thus  $b_{il} \geq \alpha(1)$ , for all  $t \geq T_0 + 2D$ . (2) follows from the fact that  $\alpha_{il}(1) = \alpha(1)$  as  $\alpha(1) \geq \mu_{il}$  for all  $t \geq T_0$ , in this case.

Observe that the evolution of  $\rho_i$  at the source of session  $i$  is similar to that of  $b_{il}$  at the origin of link  $l$ . Thus repeating the above arguments for forward rate packets leaving the source and the feedback rate packets returning to the source, we can show that  $\rho_i \geq \alpha(1)$ , for all  $t \geq T_0 + 2D$ . Thus (3) follows from the fact that  $\rho_i \geq \max_{s:i=\chi(s)} \mu_s$  for all  $t \geq T_0$ .

Now we prove (4). Since  $\rho_i \geq \alpha(1)$ ,  $b_{il} \geq \alpha_{il}(1)$ , and  $\psi_{il} \geq \alpha_{il}(1) \geq \alpha(1)$ , for all links on the path of session  $i$ , at any time after the return of the feedback rate packet sent in response to the first forward rate packet sent after  $T_0$ , the second and all subsequent forward rate packets sent after  $T_0$ , set  $f_{il}$  to values not less than  $\alpha(1)$ . Also  $f_{il} \geq \mu_{il}$  for all  $t \geq T_0$ . Thus  $f_{il} \geq \alpha_{il}(1)$  any time after the second forward rate packet reaches the origin of link  $l$ . The second forward rate packet reaches every destination in at most  $D/2$  time after its start. Thus  $f_{il} \geq \alpha_{il}(1)$ ,  $\forall t \geq (T_0 + 2.5D)^-$ .

Now we prove (5). Let  $l \in \mathcal{L}(1)$ . Consider  $t \geq (T_0 + 2.5D)^-$ . The last link  $l$  link control parameter estimation is either executed after  $T_0 + 2.5D$  or is the last such estimation before  $T_0 + 2.5D$  i.e., the last estimation for  $t < T_0 + 2.5D$ . By (4),  $f_{il}^o \geq \alpha_{il}(1)$ . We first show that  $\mathcal{M}_l \neq n(l)$ . Let  $\mathcal{M}_l = n(l)$ . By Lemma 14,  $\psi_l > \max_{i \in n(l)} f_{il}^o$ , where  $\psi_l = C_l - \sum_{i \in n(l)} f_{il}^o + \max_{i \in n(l)} f_{il}^o$  (Lemma 13).

$$\begin{aligned} \psi_l &= C_l - \sum_{i \in n(l)} f_{il}^o + \max_{i \in n(l)} f_{il}^o \\ &= \sum_{i \in n(l)} \alpha_{il}(1) - \sum_{i \in n(l)} f_{il}^o + \max_{i \in n(l)} f_{il}^o \text{ (since } l \in \mathcal{L}(1), \eta_l(1) = \alpha(1)\text{)} \\ &\leq \max_{i \in n(l)} f_{il}^o \text{ (since } f_{il}^o \geq \alpha_{il}(1) \forall i \in n(l)\text{)} \end{aligned}$$

This is a contradiction. Thus all sessions are not marked, i.e.,  $\mathcal{M}_l \subset n(l)$ . Now we show that, in fact, no session is marked. Let some session be marked.  $\mathcal{M}_l \neq \phi$ . If  $i \in \mathcal{M}_l$ ,  $f_{il}^o < \psi_l$ , by Lemma 14. Since  $f_{il}^o \geq \alpha_{il}(1)$ , this means  $\alpha(1) < \psi_l$ . Thus

$$\sum_{i \in n(l) \setminus \mathcal{M}_l} \psi_{il} = C_l - \sum_{i \in \mathcal{M}_l} f_{il}^o$$

---

<sup>§</sup>The first forward rate packet after  $T_0$  must leave the source before  $T_0 + D$ . The feedback rate packet finishes its journey before  $T_0 + 2D$

$$\begin{aligned}
 &\leq C_l - \sum_{i \in \mathcal{M}_l} \alpha_{il}(1) \text{ (since } f_{il}^o \geq \alpha_{il}(1) \forall i \in n(l)) \\
 &= \sum_{i \in n(l) \setminus \mathcal{M}_l} \alpha_{il}(1) \text{ (since } l \in \mathcal{L}(1), \eta_l(1) = \alpha(1)) \quad (42)
 \end{aligned}$$

From (42), and since  $\hat{\mu}_{il} = \mu_{il}$ ,  $\forall i \in n(l)$ ,  $\forall t \geq T_0$ ,  $\psi_{il} \leq \alpha_{il}(1)$ ,  $\forall i \in n(l) \setminus \mathcal{M}_l$ . Since  $\alpha(1) < \psi_l$ ,  $\mathcal{M}_l \subset n(l)$ , and  $\hat{\mu}_{il} = \mu_{il}$ ,  $\psi_{il} = \mu_{il}$  for all  $i \in n(l) \setminus \mathcal{M}_l$ . This means  $\sum_{i \in n(l) \setminus \mathcal{M}_l} \mu_{il} = C_l - \sum_{i \in \mathcal{M}_l} f_{il}^o$ . Since this estimation is executed at  $t \geq T_0$ , by Lemma 12,  $\mathcal{M}_l = \phi$ . Thus  $\mathcal{M}_l = \phi = n(l) \setminus \Xi_l(0)$ . Thus (5) holds for all  $t \geq (T_0 + 2.5D)^-$ .

Now we prove (6). Let  $l \in \mathcal{L}(1)$ . Let  $t \geq (T_0 + 2.5D)^-$ . The last link  $l$  link control parameter estimation is either executed after  $T_0 + 2.5D$  or is the last such estimation before  $T_0 + 2.5D$  i.e., the last estimation for  $t < T_0 + 2.5D$ . As argued in (5),  $\mathcal{M}_l = \phi \subset n(l)$ .

$$\begin{aligned}
 \sum_{i \in n(l)} \psi_{il} &= C_l \text{ (from Lemma 13)} \\
 &= \sum_{i \in n(l)} \alpha_{il}(1)
 \end{aligned}$$

Clearly from this,  $\psi_{il} = \alpha_{il}(1)$  for all  $l \in \mathcal{L}(1) \forall i \in n(l)$ ,  $t \geq (T_0 + 2.5D)^-$ . Thus first part of (6) follows. Now let  $i \in \Xi_l(0) \setminus \Xi_l(1)$ . From Lemma 10,  $\alpha(1) \geq \mu_{il}$ . Hence  $\alpha_{il}(1) = \alpha(1)$ . Thus the second part follows from the first. Both parts hold for all  $t \geq (T_0 + 2.5D)^-$ .

Now we prove (7). Let  $l \in \mathcal{L}(1)$ . Consider a forward rate packet leaving the source after  $T_0 + 2.5D$ . For all links  $l_1$  on session  $i$  path between the source and the origin of link  $l$ ,  $\mu_{il_1} \geq \mu_{il}$ . Thus  $\alpha_{il_1}(1) \geq \alpha_{il}(1)$ , for all such links  $l_1$ . Since  $f_{il_1} \geq \alpha_{il_1}(1)$ , for all  $t \geq T_0 + 2.5D$ ,  $f_{il_1} \geq \alpha_{il}(1)$  for all links  $l_1$  on session  $i$  path between the source and the origin of link  $l$ , for all  $t \geq T_0 + 2.5D$ . Also  $\rho_i \geq \max(\alpha(1), \max_{s: i=\chi(s)} \mu_s) \geq \alpha_{il}(p+1)$ . Thus the rate value of the rate packet is not less than  $\alpha_{il}(1)$  when it reaches the origin of link  $l$ . Since  $b_{il} \geq \alpha_{il}(1)$ , and  $\psi_{il} = \alpha_{il}(1)$ , this forward rate packet sets  $f_{il}$  equal to  $\alpha_{il}(1)$ . Thus every forward rate packet leaving the source after  $T_0 + 2.5D$  sets  $f_{il} = \alpha_{il}(1)$ . Since  $f_{il}$  is set by forward rate packets only,  $f_{il} = \alpha_{il}(1)$ , at all times after the first such session  $i$  forward rate packet reaches the origin of link  $l$ . Such a forward rate packet reaches the origin of a link  $l$  before  $T_0 + 4D$ . So first part of (7) holds for all  $t \geq (T_0 + 4D)^-$ . Now let  $i \in \Xi_l(0) \setminus \Xi_l(1)$ . From Lemma 10,  $\alpha(1) \geq \mu_{il}$ . Hence  $\alpha_{il}(1) = \alpha(1)$ . Thus

Thus the second part follows from the first. The second part holds for all  $t \geq (T_0 + 4D)^-$ .

Now we prove (8). Let there exist a session  $i \in \Xi_i(0) \setminus \Xi_i(1)$ . If  $l \in \mathcal{L}(1)$ , then the result follows from (7). Let  $l \notin \mathcal{L}(1)$ . If  $s \in m(i, l)$ , then  $s \in S(0) \setminus S(1)$ . By Lemma 8 there exists a saturation-link  $l_1 \in \mathcal{L}(1)$ , s.t.  $\chi(s) \in \Xi_{l_1}(0) \setminus \Xi_{l_1}(1)$  and  $\lambda_{\chi(s)l_1}(1) = \alpha(1)$ . Call the saturation link of virtual session  $s$ ,  $l_1(s)$ . In this case,  $l_1(s) \neq l$ . Let  $l_1(s)$  lie on the path of session  $i$  between the source and the origin of  $l$ . While proving (7) we have proved that any forward rate packet transmitted from the source after  $T_0 + 2.5D$ , sets  $f_{il_1(s)} = \alpha(1)$ . Thus the forward rate packet moves from the origin of  $l_1(s)$  with rate value equal to  $\alpha(1)$ . Thus this forward rate packet can set  $f_{il}$  to a value less than or equal to  $\alpha(p+1)$ . Since  $t \geq T_0 + 2.5D$ ,  $f_{il} \geq \alpha_{il}(p+1) \geq \alpha(p+1)$ . Thus any such forward rate packet sets  $f_{il}$  equal to  $\alpha(p+1)$ . Since  $f_{il}$  is set by incoming session  $i$  forward rate packets only,  $f_{il} = \alpha(1)$  always after the arrival of the first such forward rate packet. The first such forward rate packet reaches the origin of link  $l$  before  $T_0 + 4D$ . Thus  $f_{il} = \alpha(1)$ , for all  $t \geq (T_0 + 4D)^-$  in this case. Observe that this result holds if the saturation link of some virtual session  $j$ ,  $l_1(j)$  lies on the path of session  $i$  between the source of session  $i$  and the origin of link  $l$ , for some  $j \in m(i, l)$ . Let the above not hold for any  $j \in m(i, l)$ . Since  $l_1(j) \neq l$  for any  $j \in m(i, l)$ ,  $l_1(j)$  lies between the sink of link  $l$  and receiver  $j$  for all  $j \in m(i, l)$ . Since  $\chi(j) \in \Xi_{l_1(j)}(0) \setminus \Xi_{l_1(j)}(1)$ ,  $l_1(j) \in \mathcal{L}(1)$ , and  $\chi(j) = i$ , from (7), any forward rate packet sent from the source after  $T_0 + 2.5D$  sets  $f_{il_1(j)} = \alpha(1)$  and  $\psi_{il_1(j)} = \alpha(1)$  by (6). Thus  $f_{il_1(j)} \geq \psi_{il_1(j)} \geq \psi_{l_1(j)}$ . Thus a forward rate packet moves from the origin of link  $l_1(j)$  with  $u_p = 1$  and  $r_p = \alpha(1)$ . Thus a feedback rate packet returns to the origin of link  $l_1(j)$  with  $u_p = 1$ . The only way the rate value of this feedback rate packet can be greater than  $\alpha(1)$  is if the minimum rate of any  $\chi(j)$  receiver downstream is strictly greater than  $\alpha(1)$ . We know that  $\lambda_{\chi(j)l_1(j)}(1) = \alpha(1)$ . From feasibility of  $\vec{r}(1)$ ,  $\lambda_{\chi(j)l_1(j)}(1) \geq \mu_{\chi(j)l_1(j)}$ . Thus  $\alpha(1) \geq \mu_{\chi(j)l_1(j)}$ . So rate value of this feedback rate packet is less than or equal to  $\alpha(1)$ . Since  $u_p = 1$ , in the feedback rate packet,  $b_{il}$  is set to this rate value. Since  $b_{il} \geq \alpha_{il}(1) \geq \alpha(1)$ , this rate value is equal to  $\alpha(1)$ . Unless this origin is a merger point for other links of session  $i$ , clearly a feedback rate packet moves towards the source with  $u_p = 1$  and  $r_p = \alpha(1)$ . Using similar arguments, a feedback rate packet moves upstream with  $u_p = 1$  and  $r_p = \alpha(1)$  till it reaches a merger point. Now consider “first-level” merger points downstream of the origin of link  $l$ . A first level merger point is that

which satisfies the property that each of the session  $i$  paths downstream have a saturation link and there is no merger point between this merger point and such a link. Thus all feedback rate packets arrive at this merger point with  $u_p = 1$  and  $r_p = \alpha(1)$ . Thus a feedback rate packet moves upstream from this merger point, with  $u_p = 1$  and  $r_p = \alpha(1)$ . A “ $k$ th-level” merger point is a merger-point, downstream of which, there can only be  $1, 2, \dots, k - 1$ th level merger points. Applying this argument successively to “second-level,” “third-level” merger points, we can prove that a session  $i$  feedback rate packet reaches the origin of link  $l$  with  $u_p = 1$  and  $r_p = \alpha(1)$ . Thus  $b_{il} = \alpha(1)$  whenever a session  $i$  feedback rate packet arrives in response to a forward rate packet sent from source of session  $i$  after  $T_0 + 2.5D$ . Thus any forward rate packet sent after this feedback rate packet reaches the source sets  $f_{il} = \alpha(1)$ . This is because the forward rate packet arrives with rate value not less than  $\alpha(1)$  because  $f_{il_2} \geq \alpha_{il_2}(1) \geq \alpha(1)$  for all  $l_2$  on the path of session  $i$  for all  $t \geq T_0 + 2.5D$ . Also,  $\psi_{il} \geq \alpha_{il}(1) \geq \alpha(1)$  and  $b_{il} = \alpha(1)$ . Again the corresponding feedback rate packet sets  $b_{il} = \alpha(1)$ . The cycle repeats. So  $f_{il} = \alpha(1)$  always after the second forward rate packet reaches the origin of link  $l$ . The second forward rate packet transmitted after  $T_0 + 2.5D$  reaches the origin before  $T_0 + 5D$ . So, in this case  $f_{il} = \alpha(1)$  for all  $t \geq (T_0 + 5D)^-$ . In either case (position of  $l_1(j)$ s w.r.t  $l$ ),  $f_{il} = \alpha(1)$  for all  $t \geq (T_0 + 5D)^-$ .

Now we prove (9). Let  $l \notin \mathcal{L}(1)$ . Let there exist a session  $i \in \Xi_l(0) \setminus \Xi_l(1)$ . Thus  $\eta_{il}(1) \geq \eta_l(1) > \alpha(1)$ . Consider any time at and after the arrival of the second forward rate packet at the origin of link  $l$ , transmitted from the source after  $T_0 + 2.5D$ . By (8), this packet sets  $f_{il}$  equal to  $\alpha(1)$ . Since  $i \in \Xi_l(0) \setminus \Xi_l(1)$ ,  $\alpha(1) \geq \mu_{il}$  (Lemma 10). From (1),  $\psi_{il} \geq \eta_{il}(1)$  at this time. So  $\psi_{il} > \alpha(1) \geq \mu_{il} = \hat{\mu}_{il}$ . The last equality follows since  $\mu_{il} = \hat{\mu}_{il}$  for all  $t \geq T_0$ . It follows that  $\psi_l = \psi_{il} > \alpha(1)$ . Thus  $f_{il} < \psi_l$  at all times after the second forward rate packet arrives at the origin of link  $l$ . Thus any forward rate packet, starting from the second one, sets  $w_{il} = 1$ . We need to show that  $w_{il} = 1$  after the subroutine **estimate-link control parameter** is evoked. Note that  $f_{il} < \psi_l$  does not guarantee that  $w_{il} = 1$  after the routine is evoked, because sessions get unmarked during the routine. It only guarantees that every forward rate packet starting from the second one sets  $w_{il} = 1$ . Like before,  $f_{jl}^o, w_{jl}^o$  are the values of these variables just after the previous computation. Let  $\psi_l^1$  be the first estimate of the link control



parameter when the current computation is evoked. We know that

$$f_{il} = \alpha(1) \quad (43)$$

$$< \psi_l \quad (44)$$

$$w_{il} = 1 \quad (45)$$

$$w_{jl}^o = w_{jl} \text{ if } j \neq i, j \in n(l) \quad (46)$$

$$f_{jl}^o = f_{jl} \text{ if } j \neq i, j \in n(l) \quad (47)$$

Since  $i \in \Xi_l(0)$ ,  $f_{il} \geq \alpha_{il}(1) \geq \alpha(1)$  always after  $T_0 + 2.5D$  by (4). The current computation can be evoked by the second or the subsequent forward rate packet sent after  $T_0 + 2.5D$ . So the last computation before the current one must be done after  $T_0 + 2.5D$ . Thus

$$f_{il}^o \geq \alpha(1) \quad (48)$$

$$= f_{il} \text{ (from (43))} \quad (49)$$

We show that session  $i$  is marked after the execution of the subroutine. Let all sessions be marked when the subroutine is evoked. This has two subcases.

1. In the first one all sessions are marked at the end of the last estimation, i.e.,

$$w_{jl}^o = 1, \forall j \in n(l). \quad (50)$$

$$\text{Let } f_{il} = \max_{j \in n(l)} f_{jl} \quad (51)$$

From (43),  $\max_{j \in n(l)} f_{jl} = \alpha(1)$ . Thus  $f_{jl} \leq \alpha(1) \forall j \in n(l)$ . From (47),  $f_{jl}^o \leq \alpha(1) \forall j \in n(l), j \neq i$ . From (48),

$$f_{il}^o = \max_{j \in n(l)} f_{jl}^o \quad (52)$$

From (50) and (52) and Lemma 13,  $\psi_l = C_l - \sum_{u \in n(l)} f_{ul}^o + f_{il}^o = C_l - \sum_{u \in n(l), u \neq i} f_{ul}^o$ . By Lemma 14 and (50),  $f_{jl}^o < \psi_l \forall j \in n(l)$ . Thus

$$f_{jl}^o < C_l - \sum_{u \in n(l), u \neq i} f_{ul}^o \forall j \in n(l) \quad (53)$$

From (45), (46) and (50)  $w_{jl} = 1 \forall j \in n(l)$ . Thus from (51),  $\psi_l^1 = C_l - \sum_{u \in n(l)} f_{ul} + f_{il} = C_l - \sum_{u \in n(l), u \neq i} f_{ul}$ . So from (47), (49) and (53)

$f_{jl} < \psi_l^1 \forall j \in n(l)$ . So all sessions remain marked and the subroutine terminates with estimate  $\psi_l^1$  and all sessions marked. Thus  $w_{il} = 1$  after the execution of this subroutine.

$$\text{Now let } f_{il} < \max_{j \in n(l)} f_{jl} \quad (54)$$

$$\text{and } f_{il}^o = \max_{j \in n(l)} f_{jl}^o \quad (55)$$

Let  $f_{vl} = \max_{j \in n(l)} f_{jl}$ . Thus  $\psi_l^1 = (C_l - \max_{u \in n(l), u \neq v} f_{ul})$ . Also  $v \neq i$  and  $f_{vl} > f_{il}$ . From (47),  $f_{vl}^o > f_{il}$ . From (50), (55) and Lemma 13,  $\psi_l = C_l - \max_{u \in n(l), u \neq i} f_{ul}^o$ . Thus by Lemma 14,

$$f_{jl}^o < C_l - \max_{u \in n(l), u \neq i} f_{ul}^o, \forall j \in n(l). \quad (56)$$

Now  $(C_l - \sum_{u \in n(l), u \neq i} f_{ul}^o) - (C_l - \sum_{u \in n(l), u \neq v} f_{ul}) = f_{il} - f_{vl}^o < 0$ . So  $(C_l - \max_{u \in n(l), u \neq i} f_{ul}^o) < (C_l - \max_{u \in n(l), u \neq v} f_{ul}) = \psi_l^1$ . Thus  $f_{jl}^o < \psi_l^1 \forall j \in n(l)$ , from (56). From (47) and (49)  $f_{jl} < \psi_l^1, \forall j \in n(l)$ . So all sessions remain marked and the subroutine terminates with estimate  $\psi_l^1$  and all sessions marked. Thus  $w_{il} = 1$  after the execution of this subroutine.

$$\text{Now let } f_{il} < \max_{j \in n(l)} f_{jl} \quad (57)$$

$$\text{and } f_{il}^o < \max_{j \in n(l)} f_{jl}^o. \quad (58)$$

Let  $f_{zl} = \max_{j \in n(l)} f_{jl}$  and  $f_{yl}^o = \max_{j \in n(l)} f_{jl}^o$ . Clearly  $i \neq y$ , and  $i \neq z$ . Since  $f_{zl} = f_{zl}^o$ ,  $f_{zl} \leq f_{yl}^o$ . Similarly,  $f_{yl}^o \leq f_{zl}$ . Thus  $f_{zl} = f_{yl}^o$ . Thus from (47),  $f_{zl} = f_{yl}$ . Thus

$$\begin{aligned} \sum_{u \in n(l), u \neq y} f_{ul}^o &= \sum_{u \in n(l), u \neq y, u \neq i} f_{ul}^o + f_{il}^o \text{ (since } i \neq y, i \in n(l)) \\ &= \sum_{u \in n(l), u \neq y, u \neq i} f_{ul} + f_{il}^o \text{ (since from (47) } f_{ul} = f_{ul}^o \text{ if } u \neq i) \\ &\geq \sum_{u \in n(l), u \neq y, u \neq i} f_{ul} + f_{il} \text{ (since } f_{il} \leq f_{il}^o \text{ from (49))} \\ &= \sum_{u \in n(l), u \neq y} f_{ul} \end{aligned}$$

$$= \sum_{u \in n(l), u \neq z} f_{ul} \text{ (since } f_{zl} = f_{yl})$$

$$\text{Thus } C_l - \sum_{u \in n(l), u \neq y} f_{ul}^o \leq C_l - \sum_{u \in n(l), u \neq z} f_{ul} \quad (59)$$

Again from (50) and Lemma 13,  $\psi_l = C_l - \sum_{u \in n(l), u \neq y} f_{ul}^o$ . From Lemma 14,  $f_{jl}^o < \psi_l, \forall j \in n(l)$ . Thus  $f_{jl}^o < C_l - \sum_{u \in n(l), u \neq y} f_{ul}^o \forall j \in n(l)$ . Thus from (59)  $f_{jl}^o < C_l - \sum_{u \in n(l), u \neq z} f_{ul}$ . Also  $\psi_l^1 = C_l - \sum_{u \in n(l), u \neq z} f_{ul}$ . So  $f_{jl}^o < \psi_l^1$  for all  $j \in n(l)$ . From (47) and (49),  $f_{jl} < \psi_l^1$  for all  $j \in n(l)$ . So all sessions remain marked and the subroutine terminates with estimate  $\psi_l^1$  and all sessions marked. Thus  $w_{il} = 1$  after the execution of this subroutine.

2. Now let all sessions be marked before the current computation, but session  $i$  be marked because of the arrival of the current forward rate packet, i.e., session  $i$  was not marked at the end of the previous computation. Thus  $\mathcal{M}_l \subset n(l)$ . Hence  $\psi_{il} = C_l - \sum_{j \in n(l), j \neq i} f_{jl}^o$  (Lemma 13). Clearly,  $f_{il} < \psi_l$  (else session  $i$  will not be marked before this computation). Thus  $f_{il} < \psi_l \leq \psi_{il} = C_l - \sum_{j \in n(l), j \neq i} f_{jl}^o$ . Also  $f_{jl}^o = f_{jl}, \forall i \neq j$  ((47)). Thus  $f_{il} < C_l - \sum_{j \in n(l), j \neq i} f_{jl}$ . Thus  $\sum_{j \in n(l)} f_{jl} < C_l$ . Thus  $\psi_l^1 = C_l - \sum_{j \in n(l)} f_{jl} + \max_{j \in n(l)} f_{jl} > \max_{j \in n(l)} f_{jl}$ . The last inequality follows since  $\sum_{j \in n(l)} f_{jl} < C_l$ . Thus all sessions remain marked after the first step. So the subroutine terminates with all sessions marked and estimate  $\psi_l^1$ . It follows that if all sessions are marked in the beginning of the current computation, all sessions are marked at the end of the current computation. Thus session  $i$  is also marked at the end of the current computation.

Now let all sessions not be marked in the beginning of the current computation. Let  $w_{il}^o = 1$ . From (45) and (46),  $w_{jl}^o = w_{jl} \forall j \in n(l)$ . Thus  $\mathcal{M}_l = \{j : w_{jl}^o = 1\} = \{j : w_{jl} = 1\} \subset n(l)$ .

$$\begin{aligned} \sum_{j: w_{jl}^o=0} \mu_{jl} &= \sum_{j: w_{jl}^o=0} \mu_{jl} \\ &\leq \sum_{j: w_{jl}^o=0} \psi_{jl} \text{ (since } t \geq T_0, \mu_{jl} = \hat{\mu}_{jl} \forall j \in n(l)) \\ &= C_l - \sum_{j: w_{jl}^o=1} f_{jl}^o \text{ (Lemma 13)} \end{aligned}$$

$$\begin{aligned}
&\leq C_l - \sum_{j:w_{jl}^o=1} f_{jl} \text{ (from (47) and (49))} \\
&= C_l - \sum_{j:w_{jl}=1} f_{jl} \text{ (since } w_{jl} = w_{jl}^o \forall j \in n(l)) \tag{60}
\end{aligned}$$

$$\text{Let } \sum_{j:w_{jl}=0} \mu_{jl} = C_l - \sum_{j:w_{jl}=1} f_{jl}$$

Thus  $f_{jl}^o = f_{jl} \forall j$  s.t.  $w_{jl}^o = 1$ . Here  $w_{jl}^o = w_{jl} \forall j \in n(l)$ . Thus  $f_{jl} = f_{jl}^o$  for all  $j$  s.t.  $w_{jl} = 1$ . It follows that  $\sum_{j:w_{jl}^o=0} \mu_{jl} = C_l - \sum_{j:w_{jl}^o=1} f_{jl}^o$ . Since  $\mathcal{M}_l \subset n(l)$ , this means that  $w_{jl}^o = 0$  for all  $j \in n(l)$  (Lemma 12). However, we know that  $w_{il}^o = 1$  and  $i \in n(l)$ . So  $\sum_{j:w_{jl}^o=0} \mu_{jl} \neq C_l - \sum_{j:w_{jl}^o=1} f_{jl}^o$ . Thus from (60),  $\sum_{j:w_{jl}^o=0} \mu_{jl} < C_l - \sum_{j:w_{jl}^o=1} f_{jl}^o$ . Now let  $w_{il}^o = 0$ . From (45) and (46),  $\{j : w_{jl} = 0\} = \{j : w_{jl}^o = 0, j \neq i\}$ . Thus  $\mathcal{M}_l = \{j : w_{jl}^o = 1\} \subset \{j : w_{jl} = 1\} \subset n(l)$ .

$$\begin{aligned}
\sum_{j:w_{jl}=0} \mu_{jl} &\leq \sum_{j:w_{jl}^o=0, j \neq i} \psi_{jl} \text{ (since } t \geq T_0, \mu_{jl} = \hat{\mu}_{jl} \forall j \in n(l)) \\
&= C_l - \psi_{il} - \sum_{j:w_{jl}^o=1} f_{jl}^o \text{ (from Lemma 13 since } \mathcal{M}_l \subset n(l)) \\
&< C_l - f_{il} - \sum_{j:w_{jl}^o=1} f_{jl}^o \text{ (from (44))} \\
&= C_l - f_{il} - \sum_{j:w_{jl}^o=1} f_{jl} \text{ (since } w_{jl}^o = 1 \text{ means that } j \neq i \text{ and } f_{jl}^o = f_{jl} \text{ if } j \neq i) \\
&= C_l - \sum_{j:w_{jl}=1} f_{jl} \text{ (since } \{j : w_{jl} = 1\} = \{j : w_{jl}^o = 1\} \cup \{i\} \text{ and } w_{il}^o \neq 1.)
\end{aligned}$$

Thus  $\sum_{j:w_{jl}=0} \mu_{jl} < C_l - \sum_{j:w_{jl}=1} f_{jl}$  if all sessions are not marked in the beginning of the current computation. We know that session  $i$  is marked at the beginning of this computation. Let session  $i$  be marked at the end of the previous computation, i.e.,  $w_{il}^o = 1$ . From (45) and (46),  $w_{jl}^o = w_{jl}$  for all  $j \in n(l)$ . As argued before,  $\mathcal{M}_l \subset n(l)$  in this case. Thus

$$\begin{aligned}
\sum_{j:w_{jl}=0} \psi_{jl} &= \sum_{j:w_{jl}^o=0} \psi_{jl} \text{ (since } w_{jl} = w_{jl}^o \forall j \in n(l)) \\
&= C_l - \sum_{j:w_{jl}^o=1} f_{jl}^o \text{ (from Lemma 13 since } \mathcal{M}_l \subset n(l)) \\
&= C_l - \sum_{j:w_{jl}=1} f_{jl}^o \text{ (since } w_{jl} = w_{jl}^o \forall j \in n(l))
\end{aligned}$$

$$\leq C_l - \sum_{j:w_{jl}=1} f_{jl} \text{ (from (47) and (49))} \quad (61)$$

Since  $\mathcal{M}_l \subset n(l)$  and  $\sum_{j:w_{jl}=0} \mu_{jl} < C_l - \sum_{j:w_{jl}=1} f_{jl}$ ,

$$\sum_{j:w_{jl}=0} \psi_{jl}^1 = C_l - \sum_{j:w_{jl}=1} f_{jl} \quad (62)$$

$$\geq \sum_{j:w_{jl}=0} \psi_{jl} \text{ (from (61))} \quad (63)$$

$$\sum_{j:w_{jl}=0} \mu_{jl} < C_l - \sum_{j:w_{jl}=1} f_{jl} \quad (64)$$

$$\sum_{j:w_{jl}=0} \psi_{jl}^1 > \sum_{j:w_{jl}=0} \mu_{jl} \text{ (from (62) and (64))} \quad (65)$$

From (63),  $\psi_{jl}^1 \geq \psi_{jl}$ ,  $\forall j$  s.t.  $w_{jl} = 0$ . Also  $\mu_{jl} = \hat{\mu}_{jl}$ ,  $\forall j \in n(l)$ , since  $t \geq T_0$ . From (65),  $\psi_{jl}^1 > \mu_{jl}$ , for some  $j$  s.t.  $w_{jl} = 0$ . Thus  $\psi_{jl}^1 = \psi_l^1$  for such  $j$ . Thus  $\psi_l^1 = \psi_{jl}^1 \geq \psi_{jl} \geq \psi_l$ . If  $w_{jl}^o = 1$  by Lemma 14,  $f_{jl}^o < \psi_l$ . Thus  $f_{jl}^o < \psi_l^1$ , if  $w_{jl} = 1$  since  $w_{jl} = w_{jl}^o$ ,  $\forall j \in n(l)$  and  $\psi_l^1 \geq \psi_l$ . From (47) and (49),  $f_{jl} < \psi_l^1$ , if  $w_{jl} = 1$ . So the subroutine terminates with estimate  $\psi_l^1$  and without any change in the marking status of any session. Since  $w_{il} = 1$  before the subroutine is evoked,  $w_{il} = 1$  upon subroutine termination. Hence the result holds. Now let  $w_{il}^o = 0$ . As argued before,  $\mathcal{M}_l \subset n(l)$  in this case. Thus

$$\begin{aligned} \sum_{j:w_{jl}=0} \psi_{jl} &= \sum_{j:w_{jl}^o=0} \psi_{jl} - \psi_{il} \text{ (since } w_{il} = 1 \text{ and } w_{il}^o = 0) \\ &= C_l - \sum_{j:w_{jl}^o=1} f_{jl}^o - \psi_{il} \text{ (from Lemma 13 since } \mathcal{M}_l \subset n(l)) \\ &< C_l - \sum_{j:w_{jl}^o=1} f_{jl}^o - f_{il} \text{ (since } f_{il} < \psi_l \text{ from (44) and } \psi_l \leq \psi_{il}) \\ &= C_l - \sum_{j:w_{jl}^o=1} f_{jl} - f_{il} \text{ (from (47) and since } i \neq j \text{ as } w_{jl}^o = 1) \\ &= C_l - \sum_{j:w_{jl}=1, j \neq i} f_{jl} - f_{il} \text{ (since } \{j : w_{jl}^o = 1\} = \{j : w_{jl} = 1\} \setminus \{i\}) \\ &= C_l - \sum_{j:w_{jl}=1} f_{jl} \text{ (since } w_{il} = 1) \\ &= \sum_{j:w_{jl}=0} \psi_{jl}^1 \text{ (since } \mathcal{M}_l \subset n(l) \text{ and } \sum_{j:w_{jl}=0} \mu_{jl} < C_l - \sum_{j:w_{jl}=1} f_{jl}) \quad (66) \end{aligned}$$

Since  $w_{jl} = 0$  for some  $j \in n(l)$ , (by assumption all sessions are not marked) and  $\hat{\mu}_{jl} = \mu_{jl} \forall t \geq T_0$ ,  $\psi_l < \psi_l^1$ . Now  $w_{jl} = 1$  means either  $j = i$  or  $w_{jl}^o = 1$ . In the first case,  $f_{il} < \psi_l$  by (44). In the second case,  $f_{jl} < \psi_l$  by Lemma 14. Thus  $f_{jl} < \psi_l < \psi_l^1$  if  $w_{jl} = 1$ . So the subroutine terminates with estimate  $\psi_l^1$  and without any change in the marking status of any session. Since  $w_{il} = 1$  before the subroutine is evoked,  $w_{il} = 1$  upon subroutine termination. Hence the result holds always after the second packet sent after  $T_0 + 2.5D$  reaches the origin of link  $l$ . Thus  $w_{il} = 1$  for all  $t \geq (T_0 + 5D)^-$ .

Thus (1) to (9) holds for  $t \geq T_1^-$ , where  $T_1 = T_0 + 5D$ . Now we prove the last part. Consider a virtual session  $s$  s.t.  $s \in S(0) \setminus S(1)$ . Consider a forward rate packet sent from the source of session  $\chi(s)$  after  $T_0 + 2.5D$ . From Lemma 8, this forward rate packet traverses through a saturation-link  $l$  which satisfies the property that  $l \in \mathcal{L}(1)$  and  $\chi(s) \in \Xi_l(0) \setminus \Xi_l(1)$ . As argued in (7), it sets  $f_{\chi(s)l} = \alpha(1)$  at this link. Thus the forward rate packet traverses down this link with rate value equal to  $\alpha(1)$ . Note that as a forward rate packet moves downstream, rate value can only decrease or remain the same. We know that  $f_{il} \geq \alpha_{il}(1)$  for all  $t \geq T_0 + 2D$ . Thus the forward rate packet sets  $f_{il}$  to values not less than  $\alpha_{il}(1)$  and  $\alpha_{il}(1) \geq \alpha(1)$ . Thus rate values of all downstream rate packets are equal to  $\alpha(1)$ . Thus the receiver receives a forward rate packet with rate value equal to  $\alpha(1)$ . Since  $s \in S(0) \setminus S(1)$ ,  $r_s(1) = \alpha(1)$  and from feasibility of  $\vec{r}(1)$ ,  $r_s(1) \geq \mu_s$ . Thus  $\alpha(1) \geq \mu_s$ . Thus the receiver rate is set equal to  $\alpha(1)$ . This holds for all times after the first such forward rate packet reaches the receiver. The first such forward rate packet reaches the receiver before  $T_0 + 4D$ . So the corresponding receiver rate is  $\alpha(1)$  at all times after  $T_0 + 4D$ . Thus the last part holds for all  $t \geq T_1$ .  $\square$

**Lemma 16** *If for  $k = 1, \dots, p$ , there exists a time  $T_k \geq T_0$  such that  $T_1 \leq T_2 \leq \dots \leq T_k$ , and for all  $t \geq T_k^-$ ,*

1.  $\psi_{il} \geq \eta_{il}(k)$ , for all links  $l$ , and for all  $i \in \Xi_l(k-1)$ .
2.  $b_{il} \geq \alpha_{il}(k)$ ,  $\forall l, i \in \Xi_l(k-1)$ .
3.  $\rho_i \geq \max(\alpha(k), \max_{s: \chi(s)=i} \mu_s)$ ,  $\forall i$  s.t.  $i \in \Xi_l(k-1)$  for some link  $l$ .
4.  $f_{il} \geq \alpha_{il}(k)$ ,  $\forall l, i \in \Xi_l(k-1)$ .
5. If  $l \in \mathcal{L}(k)$ ,  $\mathcal{M}_l \subseteq n(l) \setminus \Xi_l(k-1)$ .

6. If  $l \in \mathcal{L}(k)$ ,  $\psi_{il} = \alpha_{il}(k)$ ,  $\forall i \in \Xi_l(k-1)$ . If  $i \in \Xi_l(k-1) \setminus \Xi_l(k)$ , and  $l \in \mathcal{L}(k)$ ,  $\psi_{il} = \alpha(k)$ .
7. If  $l \in \mathcal{L}(k)$ ,  $f_{il} = \alpha_{il}(k)$ ,  $\forall i \in \Xi_l(k-1)$ . If  $i \in \Xi_l(k-1) \setminus \Xi_l(k)$ , and  $l \in \mathcal{L}(k)$ ,  $f_{il} = \alpha(k)$ .
8. If  $i \in \Xi_l(k-1) \setminus \Xi_l(k)$ ,  $f_{il} = \alpha(k)$ .
9. If  $i \in n(l) \setminus \Xi_l(k)$ , and  $l \notin \mathcal{L}(1) \cup \mathcal{L}(2) \cup \dots \cup \mathcal{L}(k)$ ,  $w_{il} = 1$ .

and if virtual session  $s \in S(k-1) \setminus S(k)$ , then  $\rho_s = \alpha(k)$  for all  $t \geq T_k$ , then there exists a time  $T_{p+1} \geq T_p$  such that for all  $t \geq T_{p+1}^-$

1.  $\psi_{il} \geq \eta_{il}(p+1)$ , for all links  $l$ , and for all  $i \in \Xi_l(p)$ .
2.  $b_{il} \geq \alpha_{il}(p+1)$ ,  $\forall l, i \in \Xi_l(p)$ .
3.  $\rho_i \geq \max(\alpha(p+1), \max_{s: \chi(s)=i} \mu_s)$ ,  $\forall i$  s.t.  $i \in \Xi_l(p)$  for some link  $l$ .
4.  $f_{il} \geq \alpha_{il}(p+1)$ ,  $\forall l, i \in \Xi_l(p)$ .
5. If  $l \in \mathcal{L}(p+1)$ ,  $\mathcal{M}_l \subseteq n(l) \setminus \Xi_l(p)$ .
6. If  $l \in \mathcal{L}(p+1)$ ,  $\psi_{il} = \alpha_{il}(p+1)$ ,  $\forall i \in \Xi_l(p)$ . If  $i \in \Xi_l(p) \setminus \Xi_l(p+1)$ , and  $l \in \mathcal{L}(p)$ ,  $\psi_{il} = \alpha(p+1)$ .
7. If  $l \in \mathcal{L}(p+1)$ ,  $f_{il} = \alpha_{il}(p+1)$ ,  $\forall i \in \Xi_l(p)$ . If  $i \in \Xi_l(p) \setminus \Xi_l(p+1)$ , and  $l \in \mathcal{L}(p+1)$ ,  $f_{il} = \alpha(p+1)$ .
8. If  $i \in \Xi_l(p) \setminus \Xi_l(p+1)$ ,  $f_{il} = \alpha(p+1)$ .
9. If  $i \in n(l) \setminus \Xi_l(p+1)$ , and  $l \notin \mathcal{L}(1) \cup \mathcal{L}(2) \cup \dots \cup \mathcal{L}(p+1)$ ,  $w_{il} = 1$ .

and if virtual session  $s \in S(p) \setminus S(p+1)$ , then  $\rho_s = \alpha(p+1)$ ,  $\forall t \geq T_{p+1}$ .

**Proof of Lemma 16:** First we prove (1). Consider  $t \geq T_p$ . Consider a link  $l$ . If  $\Xi_l(p) = \phi$ , then (1) holds by vacuity. Let  $\Xi_l(p) \neq \phi$ . Let  $l \notin \mathcal{L}(1) \cup \mathcal{L}(2) \cup \dots \cup \mathcal{L}(p)$ . Since the last link control parameter estimation can take place at  $t \geq T_p^-$ , we know from induction hypothesis (9) that  $n(l) \setminus \Xi_l(p) \subseteq \mathcal{M}_l$ . First we assume that all sessions traversing  $l$  are not marked at the end of the last estimation, i.e.,  $\mathcal{M}_l \subset n(l)$ . Let  $\psi_l < \eta_l(p+1)$ . If  $i \in \mathcal{M}_l$ , by

Lemma 14,  $f_{ii}^o < \psi_l < \eta_l(p+1) \leq \eta_{il}(p+1)$ . If  $i \in (n(l) \setminus \Xi_l(p))$  by induction hypothesis (8) and Lemma 11,  $f_{ii}^o = \lambda_{il}(p)$ . Also let  $\mathcal{M}_l \cap \Xi_l(p) \neq \phi$ .

$$\begin{aligned}
\sum_{i \in n(l) \setminus \mathcal{M}_l} \psi_{il} &= C_l - \sum_{i \in \mathcal{M}_l} f_{ii}^o \text{ (Lemma 13)} \\
&= C_l - \sum_{i \in \mathcal{M}_l \cap \Xi_l(p)} f_{ii}^o - \sum_{i \in n(l) \setminus \Xi_l(p)} f_{ii}^o \\
&> C_l - \sum_{i \in \mathcal{M}_l \cap \Xi_l(p)} \eta_{il}(p+1) - F_l(p) \\
&\quad \text{(since } \mathcal{M}_l \cap \Xi_l(p) \neq \phi, \text{ and } f_{ii}^o < \eta_{il}(p+1), \text{ if } i \in \mathcal{M}_l \text{ and} \\
&\quad f_{ii}^o = \lambda_{il}(p) \text{ if } i \in n(l) \setminus \Xi_l(p)) \\
&= \sum_{i \in \Xi_l(p) \setminus \mathcal{M}_l} \eta_{il}(p+1) \\
&= \sum_{i \in n(l) \setminus \mathcal{M}_l} \eta_{il}(p+1) \\
&\quad \text{(since } \Xi_l(p) \setminus \mathcal{M}_l = n(l) \setminus \mathcal{M}_l \text{ as } n(l) \setminus \Xi_l(p) \subseteq \mathcal{M}_l)
\end{aligned}$$

Thus  $\psi_l > \eta_l(p+1)$ , since  $n(l) \setminus \mathcal{M}_l \neq \phi$  and  $\mu_{jl} = \hat{\mu}_{jl} \forall j \in n(l) \forall t \geq T_0$ . This is a contradiction. So if  $\mathcal{M}_l \subset n(l)$  and  $\mathcal{M}_l \cap \Xi_l(p) \neq \phi$ ,  $\psi_l > \eta_l(p+1)$ . Thus (1) holds in this case, since  $\mu_{jl} = \hat{\mu}_{jl} \forall j \in n(l)$ . Now let  $\mathcal{M}_l \cap \Xi_l(p) = \phi$  and  $\mathcal{M}_l \subset n(l)$ . Thus  $\mathcal{M}_l \subseteq n(l) \setminus \Xi_l(p)$ . Hence, from previous argument,  $\mathcal{M}_l = n(l) \setminus \Xi_l(p)$ .

$$\begin{aligned}
\sum_{i \in n(l) \setminus \mathcal{M}_l} \psi_{il} &= C_l - \sum_{i \in \mathcal{M}_l} f_{ii}^o \text{ (Lemma 13)} \\
&= C_l - \sum_{i \in n(l) \setminus \Xi_l(p)} f_{ii}^o \text{ (since } \mathcal{M}_l = n(l) \setminus \Xi_l(p)) \\
&= C_l - F_l(p) \\
&\quad \text{(since } f_{ii}^o = \lambda_{il}(p) \text{ if } i \in n(l) \setminus \Xi_l(p)) \\
\sum_{i \in \Xi_l(p)} \eta_{il}(p+1) &= C_l - F_l(p) \tag{67}
\end{aligned}$$

Also  $\Xi_l(p) = n(l) \setminus \mathcal{M}_l$  since  $\mathcal{M}_l = n(l) \setminus \Xi_l(p)$ ,  $\mathcal{M}_l \subseteq n(l)$  and  $\Xi_l(p) \subseteq n(l)$ . Since  $\mu_{jl} = \hat{\mu}_{jl} \forall j \in n(l)$ ,  $\psi_{il} = \eta_{il}(p+1)$ ,  $\forall i \in \Xi_l(p)$ . Thus (1) holds in this case,

Now let all sessions traversing  $l$  be marked at the end of the last computation, i.e.,  $\mathcal{M}_l = n(l)$ . If  $\max_{i \in n(l)} f_{ii}^o \geq \eta_l(p+1)$ , then since by Lemma 14,



$\psi_l > \max_{i \in n(l)} f_{il}^o$ ,  $\psi_l > \eta_l(p+1)$ . Now let  $\max_{i \in n(l)} f_{il}^o < \eta_l(p+1)$ . It follows that  $f_{il}^o < \eta_l(p+1)$ , for all sessions  $i$ . Now let there exist no session  $j$ , s.t.  $j \in \Xi_l(p)$ , and  $f_{jl}^o = \max_{i \in n(l)} f_{il}^o$ . This means that if  $j \in n(l)$  and  $f_{jl}^o = \max_{i \in n(l)} f_{il}^o$  then  $j \in n(l) \setminus \Xi_l(p)$ . Thus  $j \in \Xi_l(q) \setminus \Xi_l(q+1)$  for some  $q$ ,  $q < p$ . From induction hypothesis (8), this means that  $f_{jl}^o = \alpha(q+1) \leq \alpha(p)$  ((32)). Thus  $\max_{i \in n(l)} f_{il}^o \leq \alpha(p)$ . Since  $\Xi_l(p) \neq \phi$ , there exists  $j \in \Xi_l(p)$  and from assumption,  $f_{jl}^o \neq \max_{i \in n(l)} f_{il}^o$ , i.e.,  $f_{jl}^o < \max_{i \in n(l)} f_{il}^o \leq \alpha(p)$ . Since  $\Xi_l(p) \subseteq \Xi_l(p-1)$ , by induction hypothesis (4) for  $t \geq T_p$ ,  $f_{jl}^o \geq \alpha_{jl}(p) \geq \alpha(p)$ , if  $j \in \Xi_l(p)$ . This is a contradiction. So there exists a session  $j$  s.t.  $j \in \Xi_l(p)$ , and  $f_{jl}^o = \max_{i \in n(l)} f_{il}^o$ .

$$\begin{aligned}
\psi_l &= C_l - \sum_{i \in n(l)} f_{il}^o + \max_{i \in n(l)} f_{il}^o \text{ (Lemma 13)} \\
&= C_l - \sum_{i \in n(l), i \neq j} f_{il}^o \\
&= C_l - \sum_{i \in n(l) \setminus \Xi_l(p)} f_{il}^o - \sum_{i \in \Xi_l(p), i \neq j} f_{il}^o \text{ (since } j \in \Xi_l(p)) \\
&> C_l - F_l(p) - (|\Xi_l(p)| - 1) \eta_l(p+1) \\
&\quad \text{(since } f_{il} = \lambda_{il}(p) \text{ if } i \in n(l) \setminus \Xi_l(p) \text{ by induction hypothesis (8) and Lemma 11)} \\
&\quad \text{and } f_{il}^o < \eta_l(p+1), \forall i \in n(l)) \\
&= \sum_{i \in \Xi_l(p)} \eta_{il}(p+1) - (|\Xi_l(p)| - 1) \eta_l(p+1) \\
&\geq \sum_{i \in \Xi_l(p)} \eta_l(p+1) - (|\Xi_l(p)| - 1) \eta_l(p+1) \\
&= \eta_l(p+1)
\end{aligned}$$

The result follows. Now let  $l \in \mathcal{L}(q)$  for some  $q$ ,  $1 \leq q \leq p$ . Since  $i \in \Xi_l(p)$ ,  $i \in \Xi_l(q-1)$ , as  $\Xi_l(q-1) \supseteq \Xi_l(p)$ . From induction hypothesis (1),  $\psi_{il} \geq \eta_{il}(q)$ , since  $t \geq T_p \geq T_q$ . Since  $l \in \mathcal{L}(q)$ , and  $i \in \Xi_l(p)$ ,  $p \geq q$ , by Lemma 9,  $\eta_{il}(q) = \eta_{il}(p+1)$ . Thus  $\psi_{il} \geq \eta_{il}(p+1)$ , for all  $t \geq T_p$ . Thus (1) holds for all  $t \geq T_p$ .

Now we prove (2). Observe that  $\eta_l(p+1) \geq \alpha(p+1)$ , for all  $l$  if  $\Xi_l(p) \neq \phi$ . Thus  $\eta_{il}(p+1) \geq \alpha_{il}(p+1)$  for all  $i, l$  if  $\Xi_l(p) \neq \phi$ . Thus  $\psi_{il} \geq \alpha_{il}(p+1)$  for  $t \geq T_p$ , if  $i \in \Xi_l(p)$  by (1). Also observe that if  $i \in \Xi_l(p)$ , then there exists at least one receiver  $s \in m(i, l) \cap S(p)$ . For all links  $l_1$  on session  $i$  path from the origin of link  $l$  to receiver  $s$ ,  $i \in \Xi_{l_1}(p)$ . Thus  $\psi_{il_1} \geq \alpha_{il_1}(p+1)$  for all such

links for  $t \geq T_p$ . Now the argument behind (2) is similar to that behind (2) of Lemma 15, using the fact that  $\psi_{il_1} \geq \eta_{il_1}(1)$ ,  $\forall i, l_1$ . The result holds for all  $t \geq T_p + 2D$ .

Now we prove (3). If  $i \in \Xi_l(p)$ , for some  $l$ , then there exists at least one session  $i$  receiver,  $s$  such that  $s \in S(p)$ . For all links  $l \in L_s$ ,  $i \in \Xi_l(p)$  and hence  $\psi_{il_1} \geq \alpha_{il}(p+1)$  for all  $t \geq T_p$ . Now (3) can be proved in similar lines as (3) of Lemma 15. The result holds for  $t \geq T_p + 2D$ .

Using (1), (2) and (3), (4) can be argued in the same manner as (4) of Lemma 15. The result holds for  $t \geq (T_p + 2.5D)^-$ .

Now we prove (5). Let  $l \in \mathcal{L}(p+1)$ . First, let link  $l \in \mathcal{L}(q)$ , for some  $q$ ,  $1 \leq q \leq p$ . By induction hypothesis (5), for  $t \geq T_p \geq T_q$ ,  $\mathcal{M}_l \subseteq n(l) \setminus \Xi_l(q-1)$ . Since  $q \leq p$ ,  $\Xi_l(q-1) \supseteq \Xi_l(p)$ . Thus  $(n(l) \setminus \Xi_l(q-1)) \subseteq (n(l) \setminus \Xi_l(p))$ . Thus if  $t \geq T_p$ ,  $\mathcal{M}_l \subseteq (n(l) \setminus \Xi_l(p))$  after every link control parameter estimation. Now let  $l \notin \mathcal{L}(1) \cup \mathcal{L}(2) \cup \dots \cup \mathcal{L}(p)$ . If  $\Xi_l(p) = \phi$ , then (5) is trivially true. Let  $\Xi_l(p) \neq \phi$ . Let  $t \geq (T_p + 2.5D)^-$ . The last link control parameter estimation is either the last one before  $T_p + 2.5D$  or takes place after  $T_p + 2.5D$ . Thus by (4),  $f_{il}^o \geq \alpha_{il}(p+1)$ , for all  $i \in \Xi_l(p)$ . We first show that  $\mathcal{M}_l \subset n(l)$ . Let  $\mathcal{M}_l = n(l)$ . If  $i \in n(l) \setminus \Xi_l(p)$ ,  $\lambda_{il}(p) = f_{il}^o$  for all  $t \geq T_p$ , by induction hypothesis (8) and Lemma 11. Thus  $\sum_{i \in n(l) \setminus \Xi_l(p)} f_{il}^o = F_l(p)$ . By Lemma 14,  $\psi_l > \max_{i \in n(l)} f_{il}^o$ , where  $\psi_l$  is computed as  $\psi_l = C_l - \sum_{i \in n(l)} f_{il}^o + \max_{i \in n(l)} f_{il}^o$  (Lemma 13).

$$\begin{aligned}
\psi_l &= C_l - \sum_{i \in n(l)} f_{il}^o + \max_{i \in n(l)} f_{il}^o \\
&= C_l - \sum_{i \in (n(l) \setminus \Xi_l(p))} f_{il}^o - \sum_{i \in \Xi_l(p)} f_{il}^o + \max_{i \in n(l)} f_{il}^o \\
&= C_l - F_l(p) - \sum_{i \in \Xi_l(p)} f_{il}^o + \max_{i \in n(l)} f_{il}^o \\
&= \sum_{i \in \Xi_l(p)} \alpha_{il}(p+1) - \sum_{i \in \Xi_l(p)} f_{il}^o + \max_{i \in n(l)} f_{il}^o \\
&\quad (\text{since } \eta_l(p+1) = \alpha(p+1) \text{ as } l \in \mathcal{L}(p+1)) \\
&\leq \max_{i \in n(l)} f_{il}^o \text{ (since } f_{il}^o \geq \alpha_{il}(p+1) \forall i \in \Xi_l(p))
\end{aligned}$$

This is a contradiction. Thus  $\mathcal{M}_l \subset n(l)$ . Now we show that, in fact, no session in  $\Xi_l(p)$  is marked after the last computation. Let it not be so, i.e.,  $\mathcal{M}_l \cap \Xi_l(p) \neq \phi$ . If  $i \in \mathcal{M}_l$ ,  $f_{il}^o < \psi_l$  (Lemma 14). Since  $f_{il}^o \geq \alpha_{il}(p+1) \geq \alpha(p+1)$ , for  $i \in \Xi_l(p)$ , by (4), this means  $\alpha(p+1) < \psi_l$ , if  $\exists i \in \mathcal{M}_l \cap \Xi_l(p)$ ,

i.e., if  $\mathcal{M}_l \cap \Xi_l(p) \neq \phi$ . Also, since  $l \notin \mathcal{L}(1) \cup \mathcal{L}(2) \cup \dots \cup \mathcal{L}(p)$ , by induction hypothesis (9),  $(n(l) \setminus \Xi_l(p)) \subseteq \mathcal{M}_l$ . Thus  $(n(l) \setminus \Xi_l(p)) = (\mathcal{M}_l \setminus \Xi_l(p))$ .

$$\begin{aligned}
\sum_{i \in n(l) \setminus \mathcal{M}_l} \psi_{il} &= C_l - \sum_{i \in \mathcal{M}_l} f_{il}^o \text{ (from Lemma 13 since } \mathcal{M}_l \subset n(l)) \\
&= C_l - \sum_{i \in \mathcal{M}_l \cap \Xi_l(p)} f_{il}^o - \sum_{i \in \mathcal{M}_l \setminus \Xi_l(p)} f_{il}^o \\
&\leq C_l - \sum_{i \in \mathcal{M}_l \setminus \Xi_l(p)} f_{il}^o - \sum_{i \in \mathcal{M}_l \cap \Xi_l(p)} \alpha_{il}(p+1) \\
&\quad \text{(since } f_{il}^o \geq \alpha_{il}(p+1) \text{ if } i \in \Xi_l(p)) \\
&= C_l - \sum_{i \in n(l) \setminus \Xi_l(p)} f_{il}^o - \sum_{i \in \mathcal{M}_l \cap \Xi_l(p)} \alpha_{il}(p+1) \\
&\quad \text{(since } (n(l) \setminus \Xi_l(p)) = (\mathcal{M}_l \setminus \Xi_l(p)) \text{)} \\
&= C_l - F_l(p) - \sum_{i \in \mathcal{M}_l \cap \Xi_l(p)} \alpha_{il}(p+1) \\
&= \sum_{i \in \Xi_l(p)} \alpha_{il}(p+1) - \sum_{i \in \mathcal{M}_l \cap \Xi_l(p)} \alpha_{il}(p+1) \\
&\quad \text{(since } l \in \mathcal{L}(p+1), \eta_l(p+1) = \alpha(p+1)) \\
&= \sum_{i \in \Xi_l(p) \setminus \mathcal{M}_l} \alpha_{il}(p+1) \\
&\leq \sum_{i \in n(l) \setminus \mathcal{M}_l} \alpha_{il}(p+1) \text{ (since } \Xi_l(p) \subseteq n(l)) \tag{68}
\end{aligned}$$

From (68), and since  $\hat{\mu}_{jl} = \mu_{jl}, \forall j \in n(l), \psi_{il} \leq \alpha_{il}(p+1)$  for all  $i \in n(l) \setminus \mathcal{M}_l$ . However since  $\alpha(p+1) < \psi_l$   $\psi_{il} = \mu_{il}$  for all  $i \in n(l) \setminus \mathcal{M}_l$ . Thus  $\sum_{i \in n(l) \setminus \mathcal{M}_l} \mu_{il} = C_l - \sum_{i \in \mathcal{M}_l} f_{il}^o$ . Since  $\mathcal{M}_l \subset n(l)$ , and  $t \geq T_0$ , by Lemma 12,  $\mathcal{M}_l = \phi$ . Thus  $\mathcal{M}_l \cap \Xi_l(p) = \phi$ . This is a contradiction. So  $\mathcal{M}_l \cap \Xi_l(p) = \phi$ . Thus  $\mathcal{M}_l \subseteq n(l) \setminus \Xi_l(p)$ , for all  $t \geq (T_p + 2.5D)^-$ .

Now we prove (6). Let  $l \in \mathcal{L}(p+1)$ . If  $\Xi_l(p) = \phi$ , then (6) holds by vacuity. So, let  $\Xi_l(p) \neq \phi$ . Let  $l \in \mathcal{L}(q)$ , for some  $q, 1 \leq q \leq p$ . Consider a session  $i, i \in \Xi_l(p)$ . Since  $\Xi_l(p) \subseteq \Xi_l(q-1), i \in \Xi_l(q-1)$ . Thus by induction hypothesis (6),  $\psi_{il} = \alpha_{il}(q)$ , for  $t \geq T_q^-$ . We know that  $\alpha_{il}(q) \leq \eta_{il}(q)$  since  $\eta_l(q) \geq \alpha(q)$  as  $\Xi_l(q-1) \neq \phi$ . Also, by Lemma 9,  $\eta_{il}(q) = \eta_{il}(p+1)$  since  $l \in \mathcal{L}(q), i \in \Xi_l(p)$  and  $q \leq p$ . Since  $l \in \mathcal{L}(p+1), \eta_l(p+1) = \alpha(p+1)$ . Thus  $\eta_{il}(p+1) = \alpha_{il}(p+1)$ . Thus  $\psi_{il} \leq \alpha_{il}(p+1)$  for  $t \geq T_q^-$ . Since  $i \in \Xi_l(p)$ ,

we know from (1) that  $\psi_{il} \geq \alpha_{il}(p+1)$  for  $t \geq T_p$ . Also  $T_p \geq T_q^-$ . Thus for  $t \geq T_p$ ,  $\psi_{il} = \alpha_{il}(p+1)$  for all  $i \in \Xi_l(p)$ . Now let  $l \notin \mathcal{L}(1) \cup \mathcal{L}(2) \cup \dots \cup \mathcal{L}(p)$ . By induction hypothesis (9),  $(n(l) \setminus \Xi_l(p)) \subseteq \mathcal{M}_l$ , for  $t \geq T_p$ . Let  $t \geq (T_p + 2.5D)^-$ . From (5),  $\mathcal{M}_l \subseteq n(l) \setminus \Xi_l(p)$ . So  $\mathcal{M}_l = n(l) \setminus \Xi_l(p)$ . Since  $\mathcal{M}_l \subseteq n(l)$  and  $\Xi_l(p) \subseteq n(l)$ , we have  $\Xi_l(p) = n(l) \setminus \mathcal{M}_l$ . Also  $\mathcal{M}_l \subset n(l)$  as  $\Xi_l(p) \neq \phi$ .

$$\begin{aligned}
\sum_{i \in n(l) \setminus \mathcal{M}_l} \psi_{il} &= C_l - \sum_{i \in \mathcal{M}_l} f_{il}^o \text{ (by Lemma 13 since } \mathcal{M}_l \subset n(l)\text{)} \\
&= C_l - F_l(p) \text{ (since } \mathcal{M}_l = n(l) \setminus \Xi_l(p) \text{ and } f_{il}^o = \lambda_{il}(p) \text{ if } i \in n(l) \setminus \Xi_l(p)\text{,} \\
&\quad \text{by induction hypothesis (8) and Lemma 11)} \\
\sum_{i \in \Xi_l(p)} \alpha_{il}(p+1) &= C_l - F_l(p) \text{ (since } l \in \mathcal{L}(p+1)\text{)} \tag{69}
\end{aligned}$$

Since  $n(l) \setminus \mathcal{M}_l = \Xi_l(p)$ , this means that  $\psi_{il} = \alpha_{il}(p+1)$ ,  $\forall i \in \Xi_l(p)$  for  $t \geq (T_p + 2.5D)^-$ . Hence first part of (6) holds for all  $t \geq (T_p + 2.5D)^-$ . Now let  $i \in \Xi_l(p) \setminus \Xi_l(p+1)$ . From Lemma 10,  $\alpha(p+1) \geq \mu_{il}$ . Hence  $\alpha_{il}(p+1) = \alpha(p+1)$ . Thus the second part follows from the first. Both parts hold for all  $t \geq (T_p + 2.5D)^-$ .

Now we prove (7). Let  $l \in \mathcal{L}(p+1)$ . If  $\Xi_l(p) = \phi$ , then (7) holds by vacuity. So, let  $\Xi_l(p) \neq \phi$ . Now observe that, since  $i \in \Xi_l(p)$ ,  $i \in \Xi_{l_1}(p)$  for all links  $l_1$  on session  $i$  path between source of session  $i$  and the origin of link  $l$ . Thus  $f_{il_1} \geq \alpha_{il_1}(p+1)$  for all  $t \geq (T_p + 2.5D)^-$  for any such link  $l_1$  by (7). Again since  $\mu_{il_1} \geq \mu_{il}$  for any such link  $l_1$ ,  $\alpha_{il_1}(p+1) \geq \alpha_{il}(p+1)$  for any such link  $l_1$ . Also  $\rho_i \geq \max(\alpha(p+1), \max_{s: i=\chi(s)} \mu_s) \geq \alpha_{il}(p+1)$  for  $t \geq T_p + 2D$ . Thus any forward rate packet starting from the source after  $T_p + 2.5D$ , reaches the origin of link  $l$  with a rate value greater than or equal to  $\alpha_{il}(p+1)$ . By (2),  $b_{il} \geq \alpha_{il}(p+1)$  for  $t \geq T_p + 2D$ . Also  $\psi_{il} = \alpha_{il}(p+1)$  by (6) for  $t \geq T_p + 2.5D$ . So any such forward rate packet sets  $f_{il} = \alpha_{il}(p+1)$ . Since  $f_{il}$  changes only upon arrival of forward rate packets,  $f_{il} = \alpha_{il}(p+1)$  always after the arrival of the first such forward rate packet. The first such forward rate packet reaches the origin of link  $l$  by  $T_p + 4D$ . Hence  $f_{il} = \alpha_{il}(p+1)$  if  $i \in \Xi_l(p)$  for all  $t \geq (T_p + 4D)^-$ . So first part of (7) holds for all  $t \geq (T_p + 4D)^-$ . Now let  $i \in \Xi_l(p) \setminus \Xi_l(p+1)$ . From Lemma 10,  $\alpha(p+1) \geq \mu_{il}$ . Hence  $\alpha_{il}(p+1) = \alpha(1)$ . Thus the second part follows from the first. The second part holds for all  $t \geq (T_p + 4D)^-$ .

Now we prove (8). Let there exist a session  $i \in \Xi_l(p) \setminus \Xi_l(p+1)$ . If  $l \in \mathcal{L}(p+1)$ , then the result follows from (7). Let  $l \notin \mathcal{L}(p+1)$ . If  $s \in$

$m(i, l) \cap S(p)$ , then  $s \in S(p) \setminus S(p + 1)$ . By Lemma 8, virtual session  $s$  has a saturation link,  $l_1 \in \mathcal{L}(p + 1)$ , s.t.  $\chi(s) \in \Xi_{l_1}(p) \setminus \Xi_{l_1}(p + 1)$  and  $\lambda_{\chi(s)l_1}(p + 1) = \alpha(p + 1)$ . Call the saturation link,  $l_1(s)$ . In this case,  $l_1(s) \neq l$ . Let  $l_1(s)$  lie on the path of session  $i$  between the source and the origin of  $l$ . While proving (7) we have proved that any forward rate packet transmitted from the source after  $T_p + 2.5D$ , sets  $f_{il_1(s)} = \alpha(p + 1)$ . Thus the forward rate packet moves from the origin of  $l_1(s)$  with rate value equal to  $\alpha(p + 1)$ . Thus this forward rate packet can set  $f_{il}$  to a value less than or equal to  $\alpha(p + 1)$ . Since  $t \geq T_p + 2.5D$ ,  $f_{il} \geq \alpha_{il}(p + 1) \geq \alpha(p + 1)$ . Thus any such forward rate packet sets  $f_{il}$  equal to  $\alpha(p + 1)$ . Since  $f_{il}$  is set by incoming session  $i$  forward rate packets only,  $f_{il} = \alpha(p + 1)$  always after the arrival of the first such forward rate packet. The first such forward rate packet reaches the origin of link  $l$  before  $T_p + 4D$ . Thus  $f_{il} = \alpha(p + 1)$ , for all  $t \geq (T_p + 4D)^-$  in this case. Observe that this result holds if  $l_1(j)$  lies on the path of session  $i$  between the source of session  $i$  and the origin of link  $l$ , for some  $j \in m(i, l) \cap S(p)$ . Let the above not hold for any  $j \in m(i, l) \cap S(p)$ . Since  $l_1(j) \neq l$  for any  $j \in m(i, l)$ ,  $l_1(j)$  lies between the sink of link  $l$  and receiver  $j$  for all  $j \in m(i, l) \cap S(p)$ . Consider any  $j \in (m(i, l) \setminus S(p))$  (if there exists one). There exists  $q$  s.t.  $q < p$  and  $j \in S(q) \setminus S(q + 1)$ . By Lemma 8, the saturation link of  $j$ ,  $l_1(j)$  is in  $\mathcal{L}(q + 1)$ . Also  $i \in \Xi_{l_1(j)}(q) \setminus \Xi_{l_1(j)}(q + 1)$  and  $\lambda_{il_1(j)}(q + 1) = \alpha(q + 1)$ . Since  $i \in \Xi_l(p)$ ,  $i \in \Xi_{l_2}(p)$ , for all links  $l_2$  between session  $i$  source and the sink of link  $l$ . Thus  $i \in \Xi_{l_2}(q + 1)$  as  $q < p$  and hence  $\Xi_{l_2}(q + 1) \supseteq \Xi_{l_2}(p)$  for all such links  $l_2$ . Thus  $l_1(j)$  does not lie on session  $i$  path between session  $i$  source and the sink of link  $l$  for any  $j \in m(i, l) \setminus S(p)$ . Thus  $l_1(j)$  lies between the sink of link  $l$  and receiver  $j$  for all  $j \in m(i, l)$ . Let  $q(j)$  be a  $q$  s.t.  $j \in S(q) \setminus S(q + 1)$ . Thus  $\chi(j) \in \Xi_{l_1(j)}(q(j)) \setminus \Xi_{l_1(j)}(q(j) + 1)$  and  $l_1(j) \in \mathcal{L}(q(j) + 1)$ . Since  $j \in m(i, l)$  and  $i \in \Xi_l(p) \setminus \Xi_l(p + 1)$ ,  $q(j) \leq p$ . Thus  $T_{q(j)} \leq T_p$ . Thus, since  $\chi(j) \in \Xi_{l_1(j)}(q(j)) \setminus \Xi_{l_1(j)}(q(j) + 1)$ ,  $l_1(j) \in \mathcal{L}(q(j) + 1)$ , and  $\chi(j) = i$ , from (7) and induction hypothesis (7), any forward rate packet sent from the source after  $T_{q(j)} + 2.5D$  sets  $f_{il_1(j)} = \alpha(q(j) + 1)$  and  $\psi_{il_1(j)} = \alpha(q(j) + 1)$  by (6) and induction hypothesis (6). Thus  $f_{il_1(j)} \geq \psi_{il_1(j)} \geq \psi_{l_1(j)}$ . Thus a forward rate packet sent from the source after  $T_p + 2.5D$ , moves from the origin of link  $l_1(j)$  with  $u_p = 1$  and  $r_p = \alpha(q(j) + 1)$ . Thus a feedback rate packet returns to the origin of link  $l_1(j)$  with  $u_p = 1$ . The only way the rate value of this feedback rate packet can be greater than  $\alpha(q(j) + 1)$  is if the minimum rate of any  $\chi(j)$  receiver downstream is strictly greater than  $\alpha(q(j) + 1)$ . We know that  $\lambda_{\chi(j)l_1(j)}(q(j) + 1) = \alpha(q(j) + 1)$ . From feasibility of

$\bar{r}(q(j) + 1)$ ,  $\lambda_{\chi(j)l_1(j)}(q(j) + 1) \geq \mu_{\chi(j)l_1(j)}$  ((34)). Thus  $\alpha(q(j) + 1) \geq \mu_{\chi(j)l_1(j)}$ . So rate value of this feedback rate packet is less than or equal to  $\alpha(q(j) + 1)$ . Since  $u_p = 1$ , in the feedback rate packet,  $b_{il}$  is set to this rate value. Since  $b_{il} \geq \alpha_{il}(q(j) + 1) \geq \alpha(q(j) + 1)$ , (by induction hypothesis (2) and (2)), for all  $t \geq T_p + 2D$ , this rate value is equal to  $\alpha(q(j) + 1)$ . Unless this origin is a merger point for other links of session  $i$ , clearly a feedback rate packet moves towards the source with  $u_p = 1$  and  $r_p = \alpha(q(j) + 1)$ . Using similar arguments, a feedback rate packet moves upstream with  $u_p = 1$  and  $r_p = \alpha(q(j) + 1)$  till it reaches a merger point. Since  $i \in \Xi_{l_1(j)}(q(j)) \setminus \Xi_{l_1(j)}(q(j) + 1)$ ,  $y \notin S(q(j) + 1)$ ,  $\forall y \in m(i, l_1(j))$ . Thus  $q(y) \leq q(j)$ ,  $\forall y \in m(i, l_1(j))$ . By (32),  $\alpha(y(j) + 1) \leq \alpha(q(j) + 1)$ ,  $\forall y \in m(i, l_1(j))$ . Now consider “first-level” merger points downstream of the origin of link  $l$ . A first-level merger point is that which satisfies the property that each of the session  $i$  paths downstream have a saturation link and there is no merger point between this merger point and such a link. Thus a session  $i$  feedback rate packet arrives at this merger point with  $u_p = 1$  and  $r_p = \max \alpha(q(j) + 1)$ , where the maximum is taken over the session  $i$  virtual sessions traversing any session  $i$  saturation link already traversed by the feedback rate packet. Note that  $\alpha(t) \leq \alpha(t + 1)$ , for all  $t$  ((32)). Thus a feedback rate packet moves upstream from this merger point, with  $u_p = 1$  and  $r_p = \alpha(u + 1)$  where  $u$  is the largest value of  $q(j)$  for downstream session  $i$  receivers. A “ $k$ th-level” merger point is a merger-point, downstream of which, there can only be  $1, 2, \dots, k - 1$ th level merger points. Applying this argument successively to “second-level,” “third-level” merger points, we can prove that a session  $i$  feedback rate packet reaches the origin of link  $l$  with  $u_p = 1$  and  $r_p = \alpha(p + 1)$ . The last equality follows because either at least one merger point has a downstream receiver  $j$  which saturates in the  $p + 1$ th iteration, i.e.,  $j \in S(p) \setminus S(p + 1)$  or there is no merger point between the saturation link of such a receiver and the origin of link  $l$ . Thus  $b_{il} = \alpha(p + 1)$  whenever a session  $i$  feedback rate packet arrives in response to a forward rate packet sent from source of session  $i$  after  $T_0 + 2.5D$ . Thus any forward rate packet sent after this feedback rate packet reaches the source sets  $f_{il} = \alpha(p + 1)$ . This is because a forward rate packet is sent after this feedback packet, only after  $T_p + 2.5D$ . From (4)  $f_{il} \geq \alpha(p + 1)$  for all  $t \geq T_p + 2D$ , since  $i \in \Xi_l(p)$ . Thus such a forward rate packet arrives with a rate value greater than or equal to  $\alpha(p + 1)$ . Also, from (1)  $\psi_{il} \geq \alpha_{il}(p + 1) \geq \alpha(p + 1)$  and  $b_{il} = \alpha(p + 1)$ , as argued now. Thus any forward rate packet sent after this feedback rate packet sets  $f_{il} = \alpha(p + 1)$ . Again the corresponding feedback

rate packet sets  $b_{il} = \alpha(p + 1)$ . The cycle repeats. So  $f_{il} = \alpha(p + 1)$  always after the second forward rate packet reaches the origin of link  $l$ . The second forward rate packet transmitted after  $T_p + 2.5D$  reaches the origin before  $T_p + 5D$ . So, in this case  $f_{il} = \alpha(p + 1)$  for all  $t \geq (T_p + 5D)^-$ . In either case (position of  $l_1(j)$ s w.r.t  $l$ ),  $f_{il} = \alpha(p + 1)$  for all  $t \geq (T_p + 5D)^-$ .

Now we prove (9). Let  $l \notin \mathcal{L}(1) \cup \mathcal{L}(2) \cup \dots \cup \mathcal{L}(p + 1)$ . Let there exist a session  $i \in n(l) \setminus \Xi_l(p + 1)$ . There exists  $q$  s.t.  $q \leq p$  and  $i \in \Xi_l(q) \setminus \Xi_l(q + 1)$ . Thus  $T_q \leq T_p$ . Now,  $l \notin \mathcal{L}(q + 1)$ . However,  $\Xi_l(q) \neq \phi$ , since  $i \in \Xi_l(q)$ . Thus  $\eta_{il}(q + 1) \geq \eta_l(q + 1) > \alpha(q + 1)$ . Consider any time at and after the arrival of the second forward rate packet transmitted from the source after  $T_p + 2.5D$  at the origin of link  $l$ . By (8) and induction hypothesis (8),  $f_{il} = \alpha(q + 1)$ . Since  $i \in \Xi_l(q) \setminus \Xi_l(q + 1)$ ,  $\alpha(q + 1) \geq \mu_{il}$  (Lemma 10). By (1) and induction hypothesis (1),  $\psi_{il} \geq \eta_{il}(q + 1)$  at this time. So  $\psi_{il} > \alpha(q + 1) \geq \mu_{il} = \hat{\mu}_{il}$ . The last equality follows since  $t \geq T_0$ . It follows that  $\psi_l = \psi_{il} > \alpha(q + 1)$ . Thus  $f_{il} < \psi_l$  at all times after the second forward rate packet arrives at the origin of link  $l$ . Thus any forward rate packet, starting from the second one, sets  $w_{il} = 1$ . We need to show that  $w_{il} = 1$  after the subroutine **estimate-link control parameter** is evoked. As before,  $f_{jl}^o, w_{jl}^o$  are the values of these variables just after the previous computation. Let  $\psi_l^1$  be the first estimate of the link control parameter when the current computation is evoked. We know that

$$f_{il} = \alpha(q + 1) \tag{70}$$

$$< \psi_l \tag{71}$$

$$w_{il} = 1 \tag{72}$$

$$w_{jl}^o = w_{jl} \text{ if } j \neq i, j \in n(l) \tag{73}$$

$$f_{jl}^o = f_{jl} \text{ if } j \neq i, j \in n(l) \tag{74}$$

Since  $i \in \Xi_l(q)$ ,  $f_{il} \geq \alpha_{il}(q + 1) \geq \alpha(q + 1)$  always after  $(T_q + 2.5D)^-$  by (4) and induction hypothesis (4). Since  $p \geq q$ ,  $T_p \geq T_q$ . Thus  $f_{il} \geq \alpha(q + 1)$  for all  $t \geq (T_p + 2.5D)^-$ . We are considering any time after the arrival of the second forward rate packet sent after  $T_q + 2.5D$ . Thus

$$\begin{aligned} f_{il}^o &\geq \alpha(q + 1) \\ &= f_{il} \text{ (from (70))} \end{aligned} \tag{75}$$

Using (70) to (75) we can argue that the subroutine terminates after one estimate of the link control parameter and without any change in the marking

status of any session. Thus session  $i$  remains marked after the execution of the subroutine. The argument is similar to that behind (9) of Lemma 15 using (43) to (49). Hence the result holds always after the second packet sent after  $T_p + 2.5D$  reaches the origin of link  $l$ . Thus  $w_{il} = 1$  for all  $t \geq (T_p + 5D)^-$ .

Thus (1) to (9) holds for  $t \geq T_{p+1}^-$ , where  $T_{p+1} = T_p + 5D$ . Now we prove the last part. Consider a virtual session  $s$  s.t.  $s \in S(p) \setminus S(p+1)$ . Consider a forward rate packet sent from the source of session  $\chi(s)$  after  $T_p + 2.5D$ . From Lemma 8, this forward rate packet traverses through some link  $l$  which satisfies the property that  $l \in \mathcal{L}(p+1)$  and  $\chi(s) \in \Xi_l(p) \setminus \Xi_l(p+1)$ . By (7), it sets  $f_{\chi(s)l} = \alpha(p+1)$  at this link. Thus the forward rate packet traverses down this link with rate value equal to  $\alpha(p+1)$ . Note that as a forward rate packet moves downstream, rate value can only decrease or remain the same. At all subsequent links  $l_1$ , on the path to receiver  $s$ ,  $\chi(s) \in \Xi_{l_1}(p)$  since  $s \in S(p)$  and  $s \in m(\chi(s), l_1)$ . Thus the forward rate packet sets  $f_{i l_1}$  to values not less than  $\alpha_{i l_1}(p+1)$  and  $\alpha_{i l_1}(p+1) \geq \alpha(p+1)$ . Thus rate values of all downstream rate packets are equal to  $\alpha(p+1)$ . Thus the receiver receives a forward rate packet with rate value equal to  $\alpha(p+1)$ . By Lemma 10,  $\alpha(p+1) \geq \mu_{\chi(s)l} \geq \mu_s$ . Thus the receiver rate is set equal to  $\alpha(p+1)$ . This holds for all times after the first such forward rate packet reaches the receiver. The first such forward rate packet reaches the receiver before  $T_p + 4D$ . So the corresponding receiver rate is  $\alpha(p+1)$  at all times after  $T_p + 4D$ . Thus the last part holds for all  $t \geq T_{p+1}$ .  $\square$

**Proof of Theorem 3:** From Lemmas 15 and 16,  $\forall s, \sigma_s = \alpha(q(s) + 1)$ , where  $s \in S(q(s)) \setminus S(q(s) + 1)$  for all  $t \geq T_{p+1}$  if  $S(p+1) = \phi$ . We know from Theorem 3 that  $S(p+1) = \phi$ , for some  $p < M$ . Thus  $\sigma_s = \alpha(q(s) + 1)$ , for all  $t \geq T_M = t_0 + 2.5D + 5DM$ . By Lemma 11 and Theorem 3,  $\alpha(q(s) + 1)$  is the maxmin fair rate of receiver  $s$ . Thus  $\sigma_s$  is the maxmin fair receiver rate for all  $t \geq t_0 + 2.5D + 5DM$ . From Theorem 3, there does not exist a  $l$  s.t.  $i \in \Xi_l(M)$  for all sessions  $i$ . Thus from (8) of Lemmas 15 and 16 and Lemma 11  $f_{il}$ s are the maxmin fair session rates for all  $t \geq t_0 + 2.5D + 5DM$ .  $\square$

As we mentioned before, [18] presents an algorithm for computation of maxmin fair rates in unicast networks with minimum rate requirements. The authors claim that the algorithm converges in finite time after the system stabilizes. However, the proof is in error. We point out the problems in the proof. The authors prove as follows. The authors first present a centralized



algorithm for computation of maxmin fair rates. This centralized algorithm is similar to that we presented in this section, except that they consider a session saturated if it traverses a link whose capacity is fully utilized. Note that this algorithm is for unicast networks and hence virtual sessions and sessions are the same. If all sessions are unicast, then our centralized algorithm will consider a session saturated if its rate is equal to  $\alpha(k)$ , in an iteration  $k$ . Both the centralized algorithms allocate  $\max(\alpha(k), \mu_i)$  rate to unsaturated session  $i$ , for all  $i$ , in iteration  $k$ . Then the authors try to prove that the rates computed by their distributed algorithm equals that of the centralized algorithm in finite time. To show this result, they first claim that the advertized rate of a link equals  $\alpha(1)$ , for all sufficiently large  $t$ , if the link is in  $\mathcal{L}_1$  ( $\mathcal{L}(1)$  in our terminology). They also use the result that if a session saturates in iteration 1 in the centralized algorithm, then it is marked in all links except those in  $\mathcal{L}_1$  ( $\mathcal{L}(1)$  in our terminology), for all sufficiently large  $t$ . The following example shows that neither claim is true.

*Example D.1:* Consider a network with 2 links,  $e_1, e_2$  and 3 unicast sessions. Session 1 traverses both links. Session 2 traverses  $e_1$  only and session 3 traverses  $e_2$  only. Session 1 requires a minimum rate of 7. Other sessions do not have a minimum rate requirement. Link  $e_1$  has a capacity of 11 units and link  $e_2$  has a capacity of 12 units. Since sessions can enter and exit any time and link capacities can change any time during the operation of the algorithm, the marking states and session rates can be arbitrary when the network stabilizes to its current state. Let session 2 be marked in link  $e_1$  with a rate of 4 units when the network stabilizes. No other session is marked in any link at this time. The algorithm presented in [18] converges to the following advertized rates: 7 for  $e_1$  and 5 for  $e_2$ . Clearly  $\alpha(1) = 4$ . Also  $e_1 \in \mathcal{L}_1$ . The distributed algorithm generates a rate of 7 for session 1. Thus it is not marked in link  $e_2$ . Session 1 is saturated in iteration 1 and  $e_2 \notin \mathcal{L}_1$ . Thus neither claim is true.

## References

- [1] E. Amir, S. McCanne and M. Vetterli. A layered DCT coder for Internet video. *Proceedings of IEEE International Conference on Image Processing*, pages 13 – 16, Lusanne, Switzerland, September, 1996.

- [2] E. Amir, S. McCanne and H. Zhang An Application Level Video Gateway *Proceedings of ACM Multimedia' 95*, pp 255 – 265, San Francisco, CA, November 1995. ACM.
- [3] S. P. Abraham and A. Kumar. Max-Min Fair Rate Control of ABR Connections with Nonzero MCRs. *Proceedings of IEEE Globecom' 97*, pp 498 – 502
- [4] Y. Bernet, J. Binder, S. Blake, M. Carlson, B. Carpenter, S. Keshav, E. Davies, B. Ohlman, D. Verma, Z. Wang and W. Weiss. A framework for differentiated services, *Internet Draft*, February 1999.
- [5] T. Ballardie, P. Francis, and J. Crowcroft. Core based trees: an architecture for scalable inter-domain multicast routing, *Proceedings of ACM SIGCOMM*, Ithaca, NY, Sept. 1993, pp. 85-95
- [6] D. Bertsekas and R. Gallager, *Data Networks*, Englewood Cliffs, NJ:Prentice-Hall, 1987.
- [7] T. Bially, B. Gold, and S. Seneff. A technique for Adaptive Voice Flow Control in Integrated Packet Networks, *IEEE Transactions on Communications*, Vol. 28, No. 3, March 1980, pp. 325 – 333
- [8] T. Brown, S. Sazzad, C. Schoeder, P. Cantrell and J. Gibson. Packet Video For Heterogenous Network Using CU-Seeme, *Proceedings of IEEE International Conference on Image Processing*, Vol. 1, September, 1996, pp. 9 – 12.
- [9] A. Charny, An Algorithm for Rate Allocation in a Packet-Switching Network with Feedback. *M.S. thesis*, Massachusetts Institue of Technology, May 1994.
- [10] S. Chiung, M. Ammar and X. Li. On the use of Destination Set Grouping to Improve Fairness in Multicast Video Distribution *IEEE INFOCOM'96*
- [11] Z. Cao and E. Zegura. Utility Max-Min: An Application-Oriented Bandwidth Allocation Scheme *IEEE INFOCOM'99*

- [12] F. Chiussi, Y. Xia and V. Kumar. Performance of Shared-Memory Switches under Multicast Bursty Traffic *IEEE Journal on Selected Areas In Communications*, Vol 15, No. 3, 1997.
- [13] D. Cheriton and S. Deering. Host groups: A multicast extension for datagram internetworks. *Proceedings of the 9th Data Communications Symposium* (Sept. 1985), ACM/IEEE New York, 1985, 172-179
- [14] S. Deering and D. Cheriton. Multicast routing in datagram internet works and extended LANs, *ACM Transactions on Computer Systems*, vol 8, no. 2, pp. 54-60, Aug. 1994.
- [15] S. Deering, D. Estrin, D. Farinacci, V. Jacobson, C.-G. Liu, and L. Wei. An architecture for wide area multicast routing, *Proceedings of ACM SIGCOMM*, London, UK, Aug. 1994, pp. 126-135.
- [16] F. Kishino, K. Manabe, Y. Hayashi and H. Yasuda. Variable Bit-Rate Coding of Video Signals for ATM Networks, *IEEE Journal on Selected Areas In Communications*, Vol. 7, No. 5, June 1989.
- [17] M. Ghanbari. Two-Layer Coding of Video Signals for VBR Networks. *IEEE Journal on Selected Areas in Communications*, VOL. 7. No. 5. June 1989.
- [18] Y. T. Hou, H. H. Y. Tzeng and S. S. Panwar. A Generalized Max-Min Rate Allocation Policy and its Distributed Implementation Using the ABR Flow Control Mechanism, *Proceedings of IEEE Infocom' 98*, San Francisco, CA, March 1998
- [19] International Business Machines Corporation. *Technical Reference PC Network*. Doc. 6322916
- [20] X. Li, S. Paul and M. H. Ammar. Layered Video Multicast with Retransmission (LVMR): Evaluation of Hierarchical rate control, *Proceedings of IEEE Infocom' 98*, San Francisco, CA, March 1998
- [21] X. Li, S. Paul and M. H. Ammar. Multi-session Rate Control for Layered Video Multicast *Technical Report GT-CC-98–21*, College of Computing, Georgia Institute of Technology, 1998

- [22] J. Moy, Multicast routing extensions for OSPF, *Comm, ACM* vol 37, no. 8, pp 61-66, Aug 94.
- [23] Sun Microsystems. *Remote Procedure Call Reference Manual*. Mountain View, California, Oct. 1984
- [24] S. McCanne, Scalable Compression and Transmission of Internet Multicast Video. *PhD thesis*, Univ. of California, Berkeley, December 1996.
- [25] S. McCanne, V. Jacobson and M. Vetterli. Receiver-Driven Layered Multicast, *Proceedings of ACM SIGCOMM '96*, Stanford, CA, September 1996
- [26] M. Parsa, J.J.Garcia-Luna-Aceves. A protocol for Scalable Loop-Free Multicast Routing, *IEEE Journal on Selected Areas In Communications*, Vol 15, No. 3, 1997.
- [27] D. Rubenstein, J. Kurose and D. Towsley. The Impact of Multicast Layering on Network Fairness *Proceedings of ACM SIGCOMM '99, Cambridge, MA, September, 1999*
- [28] M. Satyanarayanan and F. Siegal. MultiRPC: A parallel remote procedure call mechanism. Tech Rep. CMU-CS-86-139, Carnegie-Mellon Univ., Aug. 1986
- [29] S. Sarkar and L. Tassiulas: A Framework for Routing and Congestion Control in Multicast Networks *Proceedings of IEEE INFOCOM'99*, New York, NY, March' 99
- [30] S. Sarkar and L. Tassiulas: Fair Allocation of Discrete Bandwidth Layers in Multicast Networks *Technical Report TR 99 – 43*, Institute for Systems Research and University of Maryland, 1999.
- [31] T. Turletti and J. C. Bolot Issues with multicast video distribution in heterogeneous packet networks, *Packet Video Workshop, '94*, Portland.
- [32] H. Y. Tzeng and K. Y. Siu. On Max-Min Fair Congestion Control for Multicast ABR Service in ATM, *IEEE Journal on Selected Areas In Communications*, Vol 15, No. 3, 1997.

- [33] D. L. Tennenhouse and D. J. Wetherall. Towards an active network architecture. *Computer Communication Review*, 26(2) : 5 – 18, April 1996.
- [34] D. Taubman and A. Zakhor. Multirate 3-D Subband Coding of Video, *IEEE Transactions on Image Processing*, Vol 3, No. 5, September 1994.
- [35] K. M. Uz, M. Vetterli and D. J. LeGall. Interpolative Multiresolution Coding of Advanced Television with Compatible Subchannels, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol 1, No. 1, March 1991.
- [36] M. Vishwanath and P. Chou. An efficient algorithm for hierarchical compression of video. *Proceedings of IEEE International Conference on Image Processing*, Austin, TX, November, 1994.