Multimedia Computing and Networking 1999
Volume 3654

PROCEEDINGS OF SPIE

SPIE—The International Society for Optical Engineering

Multimedia Computing and Networking 1999

Dilip D. Kandlur
Kevin Jeffay
Timothy Roscoe
Chairs/Editors

25–27 January 1999
San Jose, California

Sponsored by
IS&T—The Society for Imaging Science and Technology
SPIE—The International Society for Optical Engineering

Cosponsored by
ACM SIGMultimedia

Published by
SPIE—The International Society for Optical Engineering

P
PROCEEDINGS SERIES

Volume 3654

The papers appearing in this book comprise the proceedings of the meeting mentioned on the cover and title page. They reflect the authors' opinions and are published as presented and without change, in the interests of timely dissemination. Their inclusion in this publication does not necessarily constitute endorsement by the editors or by SPIE.

Please use the following format to cite material from this book:
Author(s), "Title of paper," in *Multimedia Computing and Networking 1999*, Dilip D. Kandlur, Kevin Jeffay, Timothy Roscoe, Editors, Proceedings of SPIE Vol. 3654, page numbers (1998).

ISSN 0277-786X
ISBN 0-8194-3125-7

Published by
SPIE—The International Society for Optical Engineering
P.O. Box 10, Bellingham, Washington 98227-0010 USA
Telephone 360/676-3290 (Pacific Time) • Fax 360/647-1445

Printed in the United States of America.

Conference Committee

Conference Chairs

Dilip D. Kandlur, IBM Thomas J. Watson Research Center
Kevin Jeffay, University of North Carolina/Chapel Hill
Timothy Roscoe, Sprint Advanced Technology Laboratories

Program Committee

H. W. Peter Beadle, Motorola Australian Research Center
Andrew Campbell, Columbia University
Shih Fu Chang, Columbia University
Ming-Syan Chen, National Taiwan University
Wu-chi Feng, The Ohio State University
Martin Freeman, Philips Research
Jose J. Garcia-Luna-Aceves, University of California/Santa Cruz
Pawan Goyal, AT&T Laboratories
Anoop Gupta, Stanford University
Mark D. Hayter, Compaq Systems Research Center
Sugih Jamin, University of Michigan
Paul Jardetzky, Aloha Networks, Inc.
Ian Leslie, University of Cambridge (UK)
Klara Nahrstedt, University of Illinois/Urbana-Champaign
Ragunathan Rajkumar, Carnegie Mellon University
Jennifer Rexford, AT&T Laboratories
Lawrence A. Rowe, University of California/Berkeley
Debanjan Saha, IBM Thomas J. Watson Research Center
Brian C. Smith, Cornell University
Cormac J. Sreenan, AT&T Laboratories
William H. Tetzlaff, IBM Thomas J. Watson Research Center
Michael Vernick, Lucent Technologies/Bell Laboratories
Harrick M. Vin, University of Texas/Austin
Jonathan Walpole, Oregon Graduate Institute of Science and Technology
Marc H. Willebeek-LeMair, IBM Thomas J. Watson Research Center
Rajendra Yavatkar, Intel Corporation
Hui Zhang, Carnegie Mellon University

Contents

SESSION 1

Conference and Session Management

Management of Large-scale Multimedia Conferencing

Israel Cidon, Youval Nachum

Department of Electrical Engineering
Technion - Israel Institute of Technology
Haifa 32000, Israel

Abstract

The goal of this work is to explore management strategies and algorithms for large-scale multimedia conferencing over a communication network. Since the use of multimedia conferencing is still limited, the management of such systems has not yet been studied in depth. A well organized and human friendly multimedia conference management should utilize efficiently and fairly its limited resources as well as take into account the requirements of the conference participants. The ability of the management to enforce fair policies and to quickly take into account the participants preferences may even lead to a conference environment that is more pleasant and more effective than a similar face to face meeting. We suggest several principles for defining and solving resource sharing problems in this context. The conference resources which are addressed in this paper are the bandwidth (conference network capacity), time (participants' scheduling) and limitations of audio and visual equipment. The participants' requirements for these resources are defined and translated in terms of Quality of Service (QoS) requirements and the fairness criteria.

A suggested solution for the problem of Capacity Resource Management (CRM) allocation is the Extended Max Min Fairness (EMMF) criterion, an extension of the well-known Max Min Fairness criterion. Both a centralized and distributed algorithms that satisfy this criterion are presented. Further tradeoffs between fairness and total throughput are also suggested. The conference time allocation problem is defined and mapped to known problems of time scheduling which are widely discussed in the literature. We examine the well-known Generalized Processor Sharing system (GPS) in that context, and select (after some adaptation) its Worst-case Fair Weighted Fair Queuing (WF^2Q) version, a scheduling policy based on the GPS system which satisfies the participants' requirements. Finally, we describe methods for combining the Time Resource Management (TRM) and the Capacity Resource Management (CRM) into a complete management of multimedia conferencing.

1 Introduction

The progress in computer networks technology accelerates the use of multimedia conferences that allow multi-level communication and collaboration between remote participants. The multimedia conference offers many advantages to the participants. It saves time and travel, avoids duplicate meetings and enables high level of collaboration.

Video conferencing over a communication network consists of multiple participants equipped with various presentation tools that communicate with each other utilizing the network services. Each participant operates his networked multimedia workstation independently of other participants. A participant may transmit to other participants a variety of information types such as real-time video streams, video and audio clips, slides and real-time shared white-board. The video conference participants are acting similar to their action in a real conference room. They may wish to listen

Part of the IS&T/SPIE Conference on Multimedia Computing and Networking 1999
San Jose, California • January 1999
SPIE Vol. 3654 • 0277-786X/98/$10.00

and watch a remote speaker, a multimedia presentation or they may wish to speak or even interrupt a speaker.

When the number of participants is small, and the participants are not greedy, the video conference management is not essential. The network capacity can be allocated in advance and kept fixed during the conference. The time management can be done by one of the participants functioning as a moderator or can be self managed using the participants well behavior and ethic. However, when the conference becomes bigger, and its participants are greedy, it must be managed by the conferencing system to guarantee a level of fairness. Otherwise, the conference will collapse in terms of its usability and perceived value. For example, a lecture with a large number of participants where all of them want to ask question at the same time. The lecturer's screen can not present all of the participants' figures simultaneously and the lecturer can not listen to all of them at the same time. Moreover, the ability to screen and control participants as well as providing fairness and take into account the participant preferences directly can make the video conference into a collaboration environment which is much better than a large face to face meeting. Management of large scale video conference can provide essential qualities that can not be provided in a conference room. The conference central management can prevent collisions, where multiple participants start speaking simultaneously or are deprived because of other participants interruptions. The presence of a conference management can guarantee the participants minimal sharing requirements. Each participant can define a set of preferences such as his minimum speaking duration during the conference and the favorite presenters (or presentations) he would like to follow. Each participant preferences are handled privately to the conference management. The participants can be preallocated guaranteed conference resources and can also interactively present new preferences during the conference. The system can take these preferences into account for its online management. For example, a favorite participant may receive a longer speaking duration than others. The conference management can secure privacy that can not be provided in a conference room. For example, each participant can specifies the participants that are allowed (or not allowed) to watch him and he can hold private discussions during a presentation without disturbing others.

The conference management should support the participants' demands by allocating its common resources to participants and other information sources. This allocation should be feasible, fair and humanly reasonable. The multi-party video conference management system faces conflicts over two main system resources, conference time and network capacity. These resources are limited by the network's restrictions and by the participants' demands. Previous related research in networked multimedia such as [1] focuses on floor control, i.e., how to allow participants of networked multimedia applications to share remote devices. The aim of floor control is to guarantee a mutually exclusive resource usage. For example, how to share a video stream. This paper examine a specific floor control problem of conference management, how to allow remote participants to collaborate and share fairly, efficiently and with no collisions the network common resources (time and capacity).

The conference time duration is a limited resource. The conference might have a planned termination time and each participant may have his own time limits. The conference Time Resource Management (TRM) function is to allocate the time resource between participants according to their demands, i.e., to determine the participants' transmission duration and order. The TRM can schedule more than one participant, so that a number of participants can transmit simultaneously to the conference. The TRM may control the participants' speaking volume. The current speaker is heard loudly and all the others at a lower volume.

The conference capacity is also a limited resource. The network capacity which is available to this conference on each network link is bounded. The Capacity Resource Management (CRM) function is to allocate capacity among all the multiple point-to-point and multicast transmissions. Capacity allocation effects the amount of concurrent sources that can be allocated in parallel as well as the video and voice quality. These limitations on the resource use lead to necessary management trade-offs, in the allocation of time and capacity among all the participants.

The conference management can be partitioned into two main management parts the TRM and the CRM. The TRM allocates time according to the source participants demands (can be related to speaker broadcast requests). The CRM allocates capacity according to the destination participants demands (can be related to destination viewing requests).

Each of resource management part is made up of two components: the admission control management and the interactive management. The admission control management checks the feasibility of the participants' requirements at the beginning of the conference and checks the online requirements of participants that wish to join during the conference. The interactive management allocates resources according to the participants' interactive demands and a priori demands.

In order to allocate resources relative to the participants' needs, each participant defines his basic service demands. These demands are based on human concepts and should be expressed in terms of Quality of Service (QoS) constraints. The use of QoS terms to define the multimedia users needs was already mentioned in [2]. In this paper we focus specifically on a multimedia conferencing, i.e., human participants take part in a large scale video conference and want to guarantee their participating rights (speak and watch during the conference).

The conference management is also affected by the conference model. There are many natural conference models, such as the "lecture model" or the "peer meeting model", that reflects meeting at daily life. We attempt to characterize several of these models and present simple examples for their implementations. A characterization of multimedia conferences was already mentioned in [3]. This paper defines the methodologies of video conference management. It describes the concepts of management related to the resources limitations and the participants needs. It also suggests specific algorithms as examples to such concepts implementations. There is still much more research needed in order to reach a full video conference management that will satisfy the participants needs and the various conference models.

In the next sections we discuss the following issues, Section 2 illustrates the main conference models, Section 3 defines the capacity allocation problem and extends the MMF allocation criterion to the Extended MMF (EMMF) criterion. Section 3.1 describes the central allocation algorithm which satisfies the EMMF criterion, and Section 3.2 describes the distributed implementation of EMMF. In order to increase the overall conference throughput, the EMMF fairness criterion is relaxed to the $\vec{\delta}$EMMF criterion as described by Appendix A. The time allocation problem and the participants' time requirements are defined by Sections 4.1 and 4.2 as a scheduling problem and are mapped to the known Generalized Processor Sharing system (GPS). This solution is extended to the Worst-case Fair Weighted Fair Queuing (WF^2Q) policy which satisfies the formalized TRM problem. Section 5 describes simple integration examples of the TRM and the CRM into an entire management system.

2 The Conference Models

Conducting large-scale video conferences over a communication network is largely a future application. It is still unknown what will be the collection of dominant applications. For our research needs we informally suggest several conference models based on common day to day models. Later, we will discuss the influence of these models on the conference management architecture.

The Peer Meeting Model. The conference participants play a similar role in the conference. They might have a different degree of information to present or different presentation capabilities. An example could be a business negotiation between several equal companies members or a standard committee meeting.

The Lecture Model. One of the conference participants is the lecturer, the rest are his audience. The lecturer is the conference speaker for most of the time and therefore employs most of its resources. The audience asks questions or makes brief remarks. They spend most of their time listening to the lecturer, or browsing through his supporting material.

The Parliament Model. Each participant may ask for two kinds of requests: Request for a speech and request for a remark. A speech is characterized by a long duration and low priority Therefore, the system response time for such a request is relatively longer. A remark is a short speaking period with high priority, therefore, the system response time for such a request is relatively shorter. There is only one participant speaking at a time (another model can assume short interruptions to speakers). The number of speaking requests for each type is limited.

The Debate Model. There is a specially-designated group of participants that addresses the conference, all the others form the audience. The audience does not speak during the conference but they interactively present their preferences, who they wish to see more. The participation order of those in the designated group and their speaking periods duration is determined interactively according to their rating. The designated participants are greedy, and spend all of the time allocated to them as, for example, in the election debate between several candidates competing for an office. Each candidate gets a period of time to explain and defend his opinions.

The Multi Group Model. This conference is characterized by working groups. A participant can be a member in more than one group at a time. He may speak and listen only to his membership groups, i.e. when one of the participants is speaking he can be seen only by the groups that he chooses from his membership groups. An example would be a large meeting of different groups from different companies that want to consult one another during the conference, without interfering other participants; they also require the privacy of not being observed by their competitors.

The Dynamic Model. With this model participants can join or leave during the conference. The number of participants is not fixed. A participant that wishes to leave can do so without informing the conference management, but participants that wish to join the conference must get approved by the conference management through an admission control policy. An example would be an Internet chat group.

3 Capacity Resource Management - (CRM)

The objective of Capacity Resource Management (CRM) is to allocate networking capacity among the participants conforming to the needs and constraints of the participants' requirements and the system capacity restrictions. Therefore, CRM should have information regarding the edge's free capacity, the information flow routing paths, and the participants' requirements. The CRM divides each edge free capacity between the specific information flows that pass through it. This division has multiple objectives that may conflict with each other. Such objectives may be maximizing the network total throughput, fair allocation, and satisfying the participants' requirements. It is assumed that participant information flows can be delivered in variable grades. For example, a real-time video can be allocated any amount of bandwidth that results in a certain grade of quality. The following sections present the main requirements of the participants, formally define participants fairness, and suggest several algorithms for given network models and given CRM objectives.

Capacity Requirements

Participant capacity requirements are derived from several factors such as the participant's applications, the participant's equipment limitations and the conference model. Video conference participants may specify their choice of sources, such as the list of sources they wish to watch and at which quality level. There are several ways in which such preferences can be presented. Destination may present to the CRM the ratio of its required allocation between the different sources. Destination may also determine for each source the maximum and the minimum rate that it is willing or able to receive. For example the minimum rate is related to the minimum video quality which the participant is willing to watch. The maximum rate maybe derived from the display equipment limitations. Destination may also restrict the total received throughput from all of the sources. The following sections focus on the destination fairness problem where each destination may only specifies a firm ratio between the sources. The network management target is to maximize the total capacity allocation of the destinations under the strict ratio constraint while utilizing the Max-Min fairness principle among the destinations.

The Network Model

Our network is modeled as a general undirected graph G(V,E). Every network component such as a switch and a router is represented by a vertex. We assume that vertices cannot fail or come up during the algorithm. $D \subseteq V$ is the destinations group. The edge (i,j)$\in$ E represents a link between two vertices of the network and C_{ij} represents the edge free capacity (available for this particular conference). We assume a reliable communication between the vertices, i.e., each message reaches its destination within a finite time. R_{sd} is an ordered set of edges that represents a single routing path from source s to destination d. Each destination may receive flow from many other sources over such routing paths. f_{sd} is the flow from source s to destination d. Every destination d determines for each source s the proportion a_{sd}. a_{sd} is the proportion of source s flow in destination d total flow f_d, i.e., f_{sd}=a_{sd} $*f_d$ and $\forall d \in D$ $\sum_{s \in V} f_{sd} = f_d$. Clearly $\forall d \in D \sum_{m \in V} a_{md} = 1$. The flow allocations are represented by a sorted flow vector $\vec{F} = \{f_{d_1}, f_{d_2}, \ldots\}$, where f_{d_1} is the smallest allocated flow.

The Extended Max Min Fairness Criterion- (EMMF)

Allocating fairly a limited capacity resource among participants requires a definition of fairness. The question is how to allocate fairly the conference capacity according to the destinations proportion requirements. The most common definition of fairness is the Max Min Fairness (MMF) [4]. The MMF solution is used in [5] for flow control fairness to provide for each flow f_{sd} a fair share. We use the formal known definition at [6] and present an extension to our conference model.

Let $\vec{F} = \{f_{d_1}, f_{d_2}, \ldots, f_{d_m}\}$ and $\vec{F^*} = \{f^*_{d_1}, f^*_{d_2}, \ldots, f^*_{d_m}\}$. We say that $\vec{F^*} \leq \vec{F}$ if vector $\vec{F^*}$ is lexicography equal or smaller than vector $\vec{F}$. Alternatively $\vec{F^*} > \vec{F}$ if $f^*_{d_1} > f_{d_1}$ or if $\forall\ j < l$ $f^*_{d_j} = f_{d_j}$ then $f^*_{d_l} > f_{d_l}$ for some $l \leq m$.

Vector $\vec{F}$ is feasible, if and only if $\forall (i,j) \in E$ $\sum_{\{s,d|(i,j)\in R_{sd}\}} f_d * a_{sd} \leq C_{ij}$

The vector $\vec{F^*} = \{f^*_{d_1}, f^*_{d_2}, \ldots\}$ *is a Max – Min fairness vector, if and only if it is feasible, and for each feasible* $\vec{F}$ $\vec{F} \leq \vec{F^*}$.

As already noted in the literature [6], the MMF approach is leaning heavily toward equal allocations of the resources, sometimes at the expense of system efficiency. In Appendix A we suggest a new criterion that relax the fairness constraint toward a maximum destination total throughput.

3.1 The Extended Max-Min Fairness Algorithm (EMMF)

Variables used by the algorithm are as follows. d - The group of destinations without capacity allocation. c_{ij} - Free residual capacity of edge (i,j). r_{ij} - The current allocated Max flow per destination in d, computed at edge (i,j).

Outline : The central algorithm computes the destination allocation vector that is EMMF optimal. It is assumed that the algorithm is provided with all the necessary network information such as routing paths and edges capacity. The algorithm first stage initializes the variables. Each edge (i,j) residual capacity c_{ij} is set to be C_{ij} the capacity of the edge. Group d, the destinations without capacity allocation, is initialized to D the group of all the destinations. The algorithm starts with calculating the maximum destinations flow allocations allowed by each edge r_{ij}, under the constraint of equal destination flow to all the non-allocated destinations (all the vertices in group d). Each edge (i,j) has its own limitation $r_{ij} = \frac{c_{ij}}{\sum_{\{m,n|(i,j)\in R_{mn} \& n \in d\}} a_{mn}}$. This means that for each flow which passes through edge (i,j) and its destination is a member of d, it allocates the maximum available flow, taking into account the local proportion of these flow allocations and the edge's residual capacity. The chosen flow is the smallest r_{ij} flow for all edges (i,j). This flow can be allocated to all the non-allocated destinations without overflowing any edge in the network. The chosen edge (m,n) is the current bottleneck edge of the network. The destinations that route their flow through the bottleneck edge (all k such that (m,n) $\in R_{vk}$)) are clearly limited by the flow allocated at this edge (r_{mn}). Therefore, r_{mn} is allocated to every such limited destination k ($f_k = r_{mn}$), and its allocated flow is subtracted from the residual free capacity of all the edges that it traverses. These participants end their role as destinations in the algorithm, and they are removed from the group of the non-allocated destinations d. The algorithm described above is repeated as long as d is not empty and uses the new values of d and c_{ij}.

Theorem 1 *The flow allocation vector $\vec{F}$, the result of the EMMF algorithm, is an EMMF vector.*

A formal statement of this algorithm along with a formal proof Theorem 1 are presented in [7].

Example:

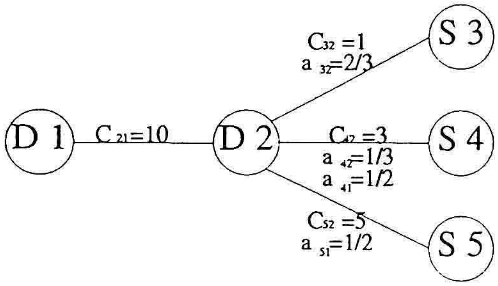

Figure 1: Optimal algorithm example

Figure 1 depicts the example graph where V = {1,2,3,4,5}, E = {(2,1),(3,2),(4,2),(5,2)} with 5 vertices all of which are participants. The capacities of the edges are $C_{21} = 10$, $C_{32} = 1$, $C_{42} = 3$, $C_{52} = 5$. The routing path from sources to destinations are $R_{41} = \{(4,2),(2,1)\}$ $R_{51} = \{(5,2),(2,1)\}$ $R_{32} = \{(3,2)\}$ $R_{42} = \{(4,2)\}$. Destination 1 determines the proportions to be $a_{41} = 1/2$, $a_{51} = 1/2$ and Destination 2 determines the proportions to be $a_{32} = 2/3$, $a_{42} = 1/3$.

The algorithm is conducted as follows. The free capacity of each edge is set to its capacity $c_{21} = 10$, $c_{32} = 1$, $c_{42} = 3$, $c_{52} = 5$. D = {1,2} has only two destinations, Participant 1 and Participant 2. The first calculation of r_{ij} results in $r_{21} = \frac{10}{1/2+1/2}$, $r_{32} = \frac{1}{2/3}$, $r_{42} = \frac{3}{1/3+1/2}$, $r_{52} = \frac{5}{1/2}$. In order to identify the bottleneck, the edge with the smallest r_{ij} is chosen. In this case, edge (3,2) is chosen as the first bottleneck and the minimum flow is $r_{32} = 3/2$. The only path that goes through edge (3,2) is the routing path between Source 3 and Destination 2. Destination 2 flow is determined to be $f_2 = 3/2$. The flow from Source 3 to Destination 2 is (3/2)*(2/3)=1, and the flow from Source 4 to Destination 2 is (3/2)*1/3=1/2. This allocated capacity is subtracted from the edges leading flow to Destination 2. $c_{32} = 1 - 1 = 0$, $c_{42} = 3 - 1/2 = 5/2$, $c_{52} = 5$. Since Destination 2 received its capacity allocation it is removed from the non-allocated group d = {1}.

Group d is still not empty so the algorithm is repeated from the first stage and calculates r_{ij} *as* $r_{21} = \frac{10}{1/2+1/2}$, $r_{32} = \infty$, $r_{42} = (5/2)/(1/2)$, $r_{52} = 5/(1/2)$. In this case we choose edge (4,2) as the bottleneck and the minimum flow is $r_{42} = 5$. The only flow that goes through the edge (4,2) is via the path between Source 4 and Destination 2. The second path that goes through the edge (4,2) has been already allocated. The current allocated destination is 1. Destination 1 flow is determined to be

$f_1 = 5$. Capacity is allocated to all the paths leading to him. The flow from Source 4 to Destination 2 is 5*(1/2)=5/2, and the flow from Source 5 to Destination 1 is 5*(1/2)=5/2. All the allocated capacity is removed from the edges, $c_{21} = 10 - 5 = 5$, $c_{42} = 5/2 - 5/2 = 0, c_{52} = 5 - 5/2 = 5/2$. Since capacity was allocated to destination 1 it is removed from the non-allocated group d. Group d is finally empty and all the destinations have received their allocations; and that leads to the end of the algorithm. The result of the algorithm is the EMMF allocation vector $\vec{F} = \{f_2 = 3/2,\ f_1 = 5\}$.

3.2 The Distributed Algorithm

Outline : The distributed algorithm computes the EMMF flow vector. The algorithm is composed of three algorithms: the source algorithm, the destination algorithm and the edge algorithm. In practice the (i,j) edges algorithm is executed at vertex i. The algorithm uses a single message format. The message includes a source identifier, a destination identifier, a minimal flow and a final mark. It is assumed that the algorithm starts simultaneously at all the destinations. Later we discuss how this requirement is relaxed.

The algorithm progresses by phases. At each phase, the destinations send the current chosen minimal flow to all their sources. All the edges over the flow paths pass this information unchanged and with no delay. At the first phase, the chosen minimal flow is set to zero and marked as "not final". The sources respond to each such destination message with a similar structured message that is sent back over the path toward the destination. At the first hop, this message carries an infinite value for the minimum flow. Before handling a source oriented message, each edge in the routing path waits until it receives all the current phase messages of the chosen minimal flow allocation (over the opposite direction) from all the destinations whose routing path goes through it. When each edge receives all such messages, it calculates its local minimal flow for that phase. The edge flow calculation is EMMF as was described for the centralized algorithm. Each edge transmits the minimum between the minimal flow that it receives from the nearest edge (down stream from the source), and its local calculated minimal flow. Each edge has its own unallocated group, a group of all source and destination pairs that do not receive their allocation. At the beginning of the algorithm this group at each edge is initiated to all the pairs that have a routing path through this specific edge. An edge marks its flow as "final" if all of its unallocated destinations choose the edge local minimum flow as their chosen minimal flow for that phase. Otherwise, the edge marks its MMF allocation as non-final. At the end of each phase, the destination receives the minimal flow over all the routing paths and chooses the minimum flow between all the paths. At this point it starts a new phase by sending this value to all its sources including the chosen flow marked as final or non-final.

Each edge in the routing path subtracts the destination chosen minimal flow from the edge capacity only when the chosen flow is marked final, i.e., it is not going to be changed. In these phases the algorithm determines the flow allocations of the next bottlenecks similar to the centralized algorithm described in Section 3.1. A bottleneck is found when an edge marks its flow as final. The algorithm terminates at a destination when its flow is going through a bottleneck edge. It receives a final status message and forwards it to all of its sources. It terminates at an edge when its unallocated group is empty, i.e., it receives a final marked flow message from all the destinations passing through it. It terminates at a source when it receives a final message from all its destinations. A formal statement of this algorithm along with a formal proof of its correctness and main properties are presented in [7].

Distributed algorithm example: Figure 2 depicts the example graph where vertices {1,2,3} are the destination group, and vertices {4,5,6,7} are the source group. Each destination has two sources with equal proportions. Destination 1 has Source 4 and Source 5. Their proportions are $a_{41} = a_{51} = 1/2$. Destination 2 has Source 5 and Source 6. Their proportion are $a_{52} = a_{62} = 1/2$. Destination 3 has source 6 and source 7. Their proportions are $a_{63} = a_{73} = 1/2$. The capacities of the edges

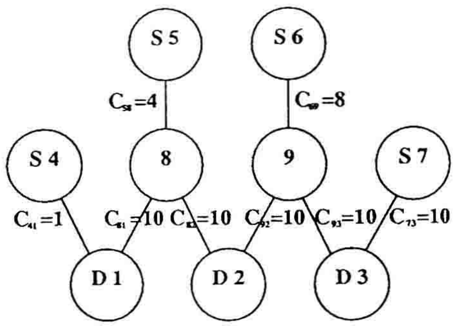

Figure 2: distributed algorithm example

are $C_{81} = C_{82} = C_{92} = C_{93} = C_{73} = 10$ and $C_{41} = 1,\ C_{58} = 4,\ C_{69} = 8$. The routing paths from sources to destinations are the shortest paths. For example $R_{41} = \{(4,1)\},\ R_{51} = \{(5,8),(8,1)\}$. The algorithm is conducted as follows: the first phase starts when all the destinations transmit their minimal chosen flow allocations set to zero. All the edges in the reverse routing paths pass this message toward the destinations without any change or delay. For each message received by the sources, each source reacts by sending a message which allows an infinite flow toward the received message specific destination. The edges in the routing path from source to destination wait until they receive all the messages from their destinations. In this phase they are all zeroes, and then calculate their MMF allocations. For example, observe the routing path {(5,8),(8,1)}. Edge (5,8) receives from its two destinations {1,2} their minimal chosen flow 0. When it receives a message that allows an infinite flow, from Source 5 to Destination 1, it passes the minimum between its calculated minimal flow 4 and the infinite flow. Edge (8,1) does the same when it receives the minimal flow from edge (5,8). It passes the minimum between its minimal calculated flow 5 and the received minimal flow 4. Destination 1 receives a minimal flow message equal to 4 on its right edge. In the second phase each destination chooses the minimum between the received flow allocations and passes it toward its sources. Destination 1 chooses 2 as the minimal flow and sends it towards sources {4,5}. In this phase, the first bottleneck is found. Edge (4,1) receives a minimal chosen flow from destination 1 equal to its allocated flow. Therefore the edge marks this flow as final. In the third phase destination 1 forwards its final flow over all its routing paths. Edge (8,5) marks Destination 1 flow as final and increases Destination 2 allocation to 6. This is repeated until all the destination flow allocations are marked final. The result of the algorithm is the EMMF allocation vector $\vec{F} = \{f_1 = 2,\ f_2 = 6, f_3 = 10\}$.

The Algorithm Main Characteristics

The algorithm avoids redundant calculations at the edges. Each edge waits for all the chosen minimal flow messages from all the routing paths that pass through it. When it has the full knowledge of all the chosen minimal flow allocations, it calculates its minimal flow for the current phase. The algorithm computation complexity is equal to the centralized algorithm. The method of spreading the information in phases reduces the number of messages that are sent in the algorithm which reduces the message complexity. The algorithm has a "fast start" feature, i.e., it allows the source to start transmitting data at the first stage. In other words, when a source receives the first chosen minimal flow message it can start transmitting. This is safe in terms of available bandwidth as is known that this flow is smaller or equal to its final (yet unknown) flow. Therefore, the sources do not overflow the edges even if they start their transmissions as soon as possible.

In the distributed algorithm we assumed that all the destinations start the algorithm at the same time. In order to relax this restriction the trigger for starting the algorithm can be performed using a Propagation of Information (PI) [8] algorithm, i.e., starting the algorithm at one vertex that floods a START message to all the destinations. Each destination starts the EMMF algorithm when it gets the START message. This change in the initializing conditions does not change the algorithm. A formal proof of this characteristic is presented in [7].

4 Time Resource Management - (TRM)

Implementing video conference over a communication network enables the control and sharing of the time resource. The Time Resource Management (TRM) point of view is a central one. It collects the timing information and requirements from all the conference participants and forces timing policies upon them. The TRM central position along with its extended communication, computation and control abilities enable it to manage the time resource better than a human moderator. It should result in time management which is more efficient (waste less time on management and better prevent collisions), precise and easy to use. In order to satisfy the above objectives,the TRM should be aware of the participants' demands. These demands can be divided into a priori demands and interactive demands. A priori demands are presented by participants before they are admitted to the conference. The interactive demands are introduced during the conference and present the current interactive participant demands. In order to satisfy these demands using a scheduling algorithm, they are presented to the TRM in terms of QoS constraints. The TRM algorithm also depends on the conference model. Several conference models were described in section 2, each of them has its own time management constraints. The objectives of the next sections are to define the main participant QoS time constraints and to map them to known QoS-based scheduling problems and solutions that were developed for other systems.

4.1 Participants Time Requirements

As described above, participants needs must be defined and specified in terms of QoS constraints. There are many types of participant applications such as real-time video, video clips, broadcast audio and human speakers. Each application may have specific QoS constraints. This section focuses on the timing requirements of human speaker participants. We present these needs in a formal and quantitative way. As mentioned before, these requirements can be divided to a priori requirements and interactive requirements. Let us start with the a priori requirements. A participant may require to speak for a time that is at least a portion of the total conference time. A participant may require to limit his waiting time from the time that he asks to speak to the time that he gets his next speaking right. Each time a participant speaks he may require not to be preempted for at least a minimum period of time before he loses his term. Before the speaker is preempted, he may wish to be alerted a specified period of time before the preemption. In conferences where participants can speak simultaneously, it may be necessary to limit the maximum number of participants speaking simultaneously at each speech level to the conference.

Although these QoS time constraints are a priori requirements, they may still be time dependent. There are cases for instance, where the opportunity to speak becomes more important as one reaches the end of the conference. At such times, more participants may want to be granted speaking rights.

We focus on what we consider to be the participant main a priori QoS time requirements. We specify them for a given speaker i.

1) ϕ_i - The minimum fraction of speaker i's total speaking periods from the sum of its waiting periods (times where i wishes to speak but has to wait) and its speaking periods. Note that if i is

greedy, i.e., wishes to speak as much as possible, his waiting periods and his speaking periods are equal to the conference duration.
2) L_i - The maximum delay from the time that a participant requests to speak till the time that he gets his speaking right.
3) P_i - Minimum speaking period without preemption.
4) M - Maximum number of participants speaking simultaneously.

We now turn to interactive requirements. A participant may also present timing requirements during the conference. He may submit a request for speaking right. When he does it, he may want to specify its characteristics. It could be a remark (a short request of high priority), or a speech (a request for a longer time). He may always stop speaking before the end of his requested speaking period. A participant may give up his time share for the sake of other participants. He may want to affect TRM decisions by presenting his preferences regarding his favorite speaker. The TRM may allocate time according to the participant's interactive rating, i.e., a participant with high rating can get a longer speaking time than a participant with a low participant rating.

We focus on two main interactive signals. 1) Requesting a speaking permission. 2) Signaling the end of speech. We will later extend these requirements.

4.2 The TRM Problem

The TRM input is the participants' QoS time requirements, the conference model, and the conference timing. The TRM is composed of the TRM admission control management and the interactive management. The interactive management is responsible for managing the conference time resource in real-time, respectively to the a priori parameters and the interactive parameters. The admission control management is responsible for checking whether the a priori input parameters are feasible or not, i.e., it checks if the scheduling algorithm can fulfill all the time constraints in the worst scenarios. Let us summarize the input parameters of the TRM considered in this section.

* The conference total time is T_{total}.
* The total number of participants is N.
* Participants may not speak simultaneously.
* The participants have a priori QoS constraints summarized in the vectors:
$\vec{\phi} = \{\phi_1, \phi_2, \ldots, \phi_N\}$, $\vec{P} = \{P_1, P_2, \ldots, P_N\}$, $\vec{L} = \{L_1, L_2, \ldots, L_N\}$.
* The participants have the following interactive requests: They can ask for a speaking permission. They can stop speaking before the termination of their allocated period.

In order to solve this TRM problem we map it to another known scheduling problem with a few changes. Our reference scheduling problem is the one that addresses link scheduling for contending packets, (see [9, 10, 11]). Here we focus on particular solution termed the Weighted Fair Queuing (WFQ) [11]. The WFQ policy is a modification of the Generalized Processor Sharing (GPS), and is also termed PGPS (Packet Generalized Processor Sharing) [10].

In the following, we briefly review the results of [10]. The GPS server serves packets from N different sessions. Each packet has an arbitrary length with a maximum of P_{max}. The server operates at a fixed rate $r = 1$. GPS is a work-conserving server, i.e., it must serve if there are packets waiting for service. Each session i is characterized by a positive number ϕ_i such that $\sum_j \phi_j \leq 1$. A session is backlogged if there are packets waiting to be processed. Let $S_i(\tau, t)$ be the amount of session i traffic served in an interval (τ, t), and let B(t) be the group of sessions being served under GPS at time t. The GPS provides for any backlogged session i in time interval of (τ, t) a fraction ϕ_i such that $\frac{\phi_i}{\phi_j} \leq \frac{S_i(\tau,t)}{S_j(\tau,t)}$ $\quad j = 1, 2, \ldots, N$. At any given time t, session i is guaranteed a rate of