REIHE INFORMATIK
6/2000

# The Fairness Challenge in Computer Networks

Robert Denda
Praktische Informatik IV
Universität Mannheim
L15, 16
D-68131 Mannheim

**Abstract**

In this paper, the concept of fairness in computer networks is investigated. We motivate the need of examining fairness issues by providing example future application scenarios where fairness support is needed in order to experience sufficient service quality. Fairness definitions from political science and their application to computer networks are described and a state-of-the-art overview of research activities in fairness, from issues such as queue management and tcp-friendliness to issues like fairness in layered multi-rate multicast scenarios, is given. We contribute with this paper to the ongoing research activities by defining the *fairness challenge* with the purpose of helping direct future investigations to the white spots on the map of research in fairness.

## 1 Introduction

Investigating *fairness* in computer network aims at two goals. The first one is to improve the behaviour of networking architectures by adding a valuable concept that should be considered both for existing and for new scenarios, and the second one is to enable new (fair) applications that are currently not implemented in existing networks for several reasons. In this paper, we will point out the necessary aspects of both goals of fairness. Both flavours of fairness follow a general abstract definition borrowed from political economics, which is explained later in Section 2.

In the following, let us first present issues concerning the second flavour of fairness: new (fair) application scenarios that require a certain level of fairness.

Current computer networks fulfill most needs of their users. Email, file transfer, WWW access, IP-telephony, video-conferencing, etc. are widely deployed and more or less well supported by most computer networks. Nevertheless, there exist real-time application scenarios that are yet not implemented, examples of which are tele stock trading (we think of intra-day stock trading from home), large-scale distributed real-time games, real-time tele auctions, etc. We believe that the main reasons for these applications not to be deployed are insufficient existing networking mechanisms.

At an abstract level, the needs of users in computer networks can be structured hierarchically as a pyramid (see Figure 1), which can be somewhat likened to Maslow's pyramid of human needs[1].

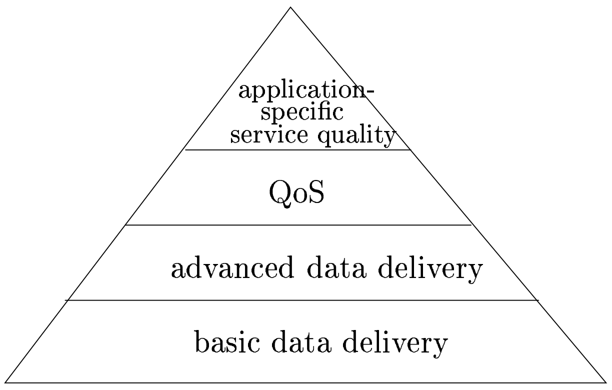

*Figure 1: Pyramid of users' needs in computer networks.*

The minimum level of users' needs in a computer network is *basic data delivery functionality*, which provides asynchronous off-line data delivery with no explicit requirements on the time or duration of delivery. At a service level, simple email service can be regarded as an example.

The next higher level in the pyramid adds *advanced data delivery functionality*: users want to be provided with synchronous, interactive, and/or two-way on-line data delivery. Examples of such services are file transfer and WWW. Note that at this level, there are still no hard bounds on the time, duration and delay of data delivery.

The third level adds *quality of service (QoS)*. Users want to perform real-time multimedia communication involving data streaming for audio/video. Therefore, they need a communication system that offers sufficient quality of service for the traffic. Note that in this context, quality of service means abstract user requirements concerning the data delivery and does not necessarily mean that the communication system needs explicit QoS support: even in well-provisioned best-effort networks users might be content with the quality of service they receive, while the network itself has no explicit mechanisms for providing QoS. Examples of communication that need a certain level of quality of service are telephony, video-conferencing, tele-education, tele-medicine, and many more.

At the highest level of the pyramid, the technical feasibility for most applications is already assured through the lower three levels. Nevertheless, there are application scenarios where mechanisms and guarantees are needed that are beyond current technology. Such application scenarios include real-time tele-stock trading, tele-voting, large-scale distributed games and real-time electronic auctions. For these applications, quality of service provision in a network alone does not suffice: for example, users want not only to be sure that the communication system provides the service with its required QoS guarantees, they also want to experience a certain application-specific *service quality*[1], in particular, a guarantee that they have at least the same opportunities as their competitors. For these applications, fairness is needed to provide the necessary service quality to the application. For that reason, *fairness* is one main aspect of the highest level of the pyramid. In addition to fairness, other needs, such as network availability, security, etc. are also located at the highest level. Note that especially due to the heterogeneity of current networks, such application-specific service quality cannot be provided through QoS guarantees only.

The lower three levels, which are necessary to make network services functional,

[1] We distinguish between *service quality*, which describes the quality that a user experiences when using a service, and *quality of service (QoS)*, which concerns network mechanisms providing a necessary level of quality of data transport in a network.

have been addressed in detail in literature and are still worthwhile a lot of further discussion (to which this paper will not contribute). Unfortunately, high-level concepts to provide the actual service quality for many applications, in particular fairness concepts in computer networks, have not been examined as thoroughly. Especially, it seems that an overall view of fairness concepts is missing. This paper is intended to fill this gap and to shed some light on the importance of fairness concepts for computer networks.

The following parts of the paper are organized as follows. Section 2 borrows models from political science to give both a common sense definition and a formal description of the concept of *fairness* at an abstract level. In addition, it presents a number of fairness definitions that are commonly used in various scenarios, and can be applied to both fairness for existing computer networks and to fair applications.

In Section 3, the general fairness concept is applied to computer networks, and existing approaches to integrate fair mechanisms into computer networks are presented.

Section 4 provides an overview and discussion of open issues and challenges in fairness research for computer networks, thereby defining the *fairness challenge* with various facets that should be investigated within future research.

Finally, Section 5, gives a short summary and concludes this paper.

## 2 What is Fairness ?

Fairness is an important and general concept and a requirement for overall happiness. In the following, we investigate the difficult question of how to properly define the term *fairness* in general and in specific to the computer network scenario.

The concept of fairness has been studied in various scientific areas. Most thorough and theory-based approaches arose from the field of political science and political economics: fair allocations of consumption bundles in an economy have been investigated, and a common sense definition of a fair allocation is given as "an allocation where no person in the economy prefers anyone else's consumption bundle over his own"[2], i.e., "a fair allocation is free of envy"[3]. Even this very general definition indicates the conceptual difficulty of fairness: in order to ensure fairness in a system, all system entities have to be satisfied with their allocated share of the system's goods. Therefore, the distribution mechanism has to take into account the subjective preferences of the system entities.

The famous problem of how to divide a cake fairly into pieces for a number $n$ of hungry and competing cake eaters has been examined in various early studies in the field of econometrics (see, e.g., [4], [5] and [6]). The cake division problem illustrates well the difficulty of the fairness concept for simple distribution problems

An algorithm that solves the cake division problem proposed by Banach and Knaster (see [4]) is very simple: a knife is moved at constant speed over the cake and is poised at each instant, s.t. it could cut a unique slice of the cake. Thus, the potential slice increases monotonely until it becomes the entire cake. The first person to indicate satisfaction with the slice determined by the position of the knife receives that slice (if two persons indicate satisfaction simultaneously, the slice is given to any one of them). Then, the rest of the cake is distributed using the same constructive method. It should be noted that any algorithm that solves the cake division problem requires active participation of the cake eaters, i.e., they have to signal their preferences to make sure that they are content with the outcome.

In computer networks, the situation is very similar: resources have to be distributed among competing users of the computer network, which can be likened to distributing pieces of cake to competing cake eaters. The practical problem with applying the cake division algorithm to resources in computer networks is that it would require active signaling of the users' preferences upon all changes of resource distribution in the network. For scalability reasons, this approach is clearly not feasible.

In order to avoid this problem of continuous signaling, a concept to express the preferences of a user has been developed: *utility functions*. Utility functions are instruments to form the base for a mathematical analysis and description of the fairness problem, and are both helpful in political science and theory of computer networking.

In the field of political economics, there exists a relatively thorough theory for the concept of fairness and related issues, all based on the concept of utility functions. After briefly reviewing the concept of utility functions, an abstract definition of fairness and a brief description of the related political economic terms *pareto efficiency* and *welfare* are given in the remainder of this section. In addition, other more concrete fairness criteria and measures that can be directly applied to computer networks are presented.

## 2.1 Utility Functions

Utility functions[2] are a means to describe preferences, by assigning numbers to all possible consumption bundles, thereby defining a relative order between these bundles. That is, a bundle $x_1$ is prefered to a bundle $x_2$ if and only if the utility $u$ of $x_1$ is larger than the utility of $x_2$ [7]. Thus, a utility function is a way to represent preference ordering. According to this definition, the numerical magnitudes of utility levels have no intrinsic meaning; therefore, any monotonic transformation of a utility function will represent the same preferences.

In order to analyze and formalize computer networks, utility functions are defined for networking applications as functions that map a service delivered by the network into the performance of the application for that service [8]. The service describes all

[2] An introduction to utility can, for example, be found in [7], Chapter 4.

relevant measures, such as delay, throughput, loss rate, etc. In this context, unlike with the definition in political economics, the magnitudes of utility levels have a meaning, since they also describe the relative differences in application performance between distinct levels of service. Usually, in computer networks, a further restriction is imposed to utility functions: *transferability*; i.e., the magnitudes of utility functions of different applications can be compared and quantities like the sums of utilities make sense. The utility can then be considered a measure of how much a user would be willing to pay for the service [8]. With this definition, utility functions in computer networks can be considered a specific subset of the utility functions of political economics, which allows for better applicability to network resource distribution problems.

Figures 2 and 3 illustrate qualitative utility functions concerning the service parameter bandwidth for the different application types *elastic applications* and *real-time applications.* Shenker[8] provides other qualitative examples of utility functions for different types of applications concerning the service parameter bandwidth. Note that the utility function illustrated in Figure 3 refers to a very similar case as the utility function for rate-adaptive applications presented by Shenker, except for the fact that Shenker assumes rate-adaptive applications to be also always delay-adaptive.

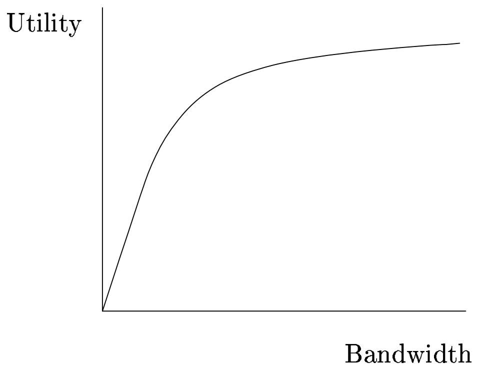

*Figure 2: Utility function for elastic applications.*

Figure 2 shows a qualitative utility function for elastic applications. Elastic applications are traditional data applications, such as file transfer or remote terminals. Elastic applications are tolerant of delays and can adapt their data rate according to the current network situation. As presented in Figure 2, for elastic applications it is typical that the value of additional bandwidth decreases with higher bandwidth. That is due to the rate adaptivity of elastic applications: they can operate at all levels of non-zero bandwidth, and an additional kilobit of data per second that can be transmitted is worth more when the current bandwidth is low than when it is already high.

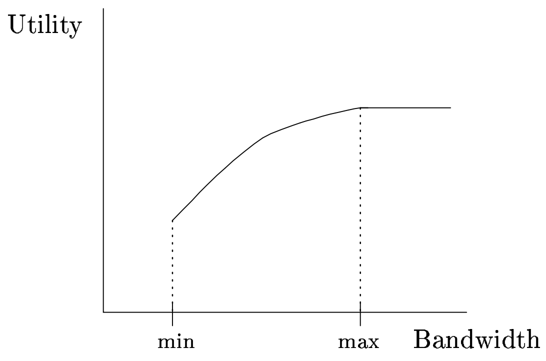

*Figure 3: Utility function for real-time applications.*

Real-time applications are applications that perform streaming of real-time data (i.e., usually audio and video), such as telephony or video-conferencing applications. Figure 3 shows the utility function of bandwidth for real-time applications. Note that for real-time applications, utility for a certain level of service represents the perceived quality of the media stream. That is, in order to correspond to the definition of utility given, we define the performance of a real-time application to be the perceived quality of the delivered media stream.

Note also that with the utility function presented in Figure 3, it is assumed that real-time applications are quite delay-sensitive and can therefore only perform little adaptation to changes in network delay. On the other hand, real-time applications can usually perform adaptation of their data-rate (by scaling the transmitted media stream) and are therefore rate-adaptive. As Figure 3 illustrates, there exists a minimum level of service below which the real-time application cannot operate. Also, note that there is a maximum level of service, above which any increase in level of service does not lead to additional utility. That is due to the fact that the codecs of real-time applications have a maximum output rate, which cannot be exceeded. Between these two levels of service, we can assume a concave function, since the higher the level of service already, the less effect an additional increase has on the quality perceived.

## 2.2 Pareto-Efficiency, Welfare and Fairness

The concepts *pareto-efficiency* and *welfare* in political economics are strongly related to the concept of *fairness* and will be briefly revised to provide a more theoretical definition of *fairness*.

Let us follow [9] to define *pareto-efficiency*:

In general, an allocation $\omega$ of resource bundles $(x_1, ..., x_k)$ is *feasible* if the *excess demand* $z(\omega)$ for that allocation is $\leq 0$. The *excess demand* is the aggregate vector of demands reduced by the aggregate vector of resources available; thus, an allocation is *feasible*, if the aggregate supply of resources exceeds or equals the aggregate resource requirements of users.

A *utility allocation* $u_i$ represents user $i$'s utility of an allocation $\omega$ for a resource bundle $(x_1, ..., x_k)$. A utility allocation $u_i^1$ is *dominated* by $u_i^2$, if $u_i^2$ is feasible and $u_i^2 > u_i^1$, i.e., if $u_i^2$ is prefered to $u_i^1$.

A utility allocation $u_i^1$ is *pareto-efficient*, if it is *feasible* and *not dominated* by any other feasible utility allocation $u_i^2$. In more general terms, "a situation is *pareto-efficient*, if there is no way to make any person better off without hurting anybody else" (see [7], Section 16.9).

Pareto-efficiency is clearly a desirable criterion of an allocation. Nevertheless, it is only a weak criterion. The problem is that also an allocation, where one user gets everything can be pareto-efficient, and this allocation is certainly not fair.

*Welfare* extends the concept of pareto-efficiency in a certain manner: the basic problem of *welfare* (see, e.g., [10]) is to determine, which of the feasible allocations $\omega(x_1, ..., x_k)$ should be selected. For that reason, it is assumed that there exists a general *welfare function* $W(u_1, u_2, ..., u_n)$ that aggregates the individual utility functions $u_i$ of the users. A welfare function is required to be increasing in all of its arguments.

It can be shown that any feasible allocation of maximum welfare must necessarily be pareto-efficient[3]. For that reason, it seems to be very desirable to find an appropriate welfare function and perform the maximization in order to receive maximum welfare while being pareto-efficient. The problem with this approach is the welfare function itself, since it is not clear how to perfectly aggregate individual preferences.

In fact, Arrow's Impossibility Theorem[12] shows that there is no perfect way to aggregate individual preferences to make one social preference, if the following desirable three properties a) through c) hold (see, e.g., [7]): a) Given any set of complete, reflexive, and transitive individual preferences, the social decision mechanism should result in social preferences that satistfy the same properties. b) If everybody prefers alternative x to alternative y, then the social preferences should rank x ahead of y. c) The preferences between x and y should depend only on how people rank x versus y, and not on how they rank other alternatives.

Nevertheless, for different purposes, different examples of welfare functions exist, each corresponding to a different criterion of welfare. For an introductory overview and comparison of different criteria of welfare see [13].

[3] For a simple proof, see, e.g., [11].

One criterion is the *maximin criterion*, which corresponds to the Rawlsian welfare function $W(u_1, ..., u_n) = min(u_1, ..., u_n)$. The maximin criterion weighs only the utility of the worst-off user. Note that this criterion requires ordinal comparisons of utilities.

The *sum of utilities criterion* corresponds to the classical utilitarian welfare function $W(u_1, ..., u_n) = \sum_i u_i$. In contrast to the maximin criterion, this criterion weighs the utility of each user equally. Again, note that this criterion involves cardinal interpersonal comparisons of utility.

Both these criteria have certain problems: the maximin criterion does not weigh improvements of those who are not least well off; and the sum of utilities criterion might prefer a situation where some users are very happy and others are very miserable, rather than allowing an allocation where all users are just happy, i.e., in between extremely happy and very miserable.

These two criteria can be regarded as the limiting cases. In between, there exist a whole range of various compromise welfare functions all aiming at different goals. One example is the *weighted-sum-of-utilities welfare* function $W(u_1, ..., u_n) = \sum_i a_i u_i$, where $a_i$ is a weight assigned to $u_i$, thereby expressing individual priorities between different users. Another example is the *sum-of-square-roots* function $W(u_1, ..., u_n) = \sum_i \sqrt{u_i}$, where users with smaller utilities are given higher relative priorities.

Yet another, but very interesting welfare function is the *sum-of-logs* function $W(u_1, ..., u_n) = \sum_i log(u_i)$, which corresponds to the so-called *Nash criterion.* Note that the *Nash criterion* maximizes the product of additional utilities compared to the status quo. It has been first described by Nash[14] as the solution to the bargaining game in game theory.

This maximized welfare function has the property that its outcome is not affected by any linear transformation of a user's utility scales: if a user's utility function is transformed using a positive linear transformation, the solution to maximizing the welfare function yields an allocation which is identical to the allocation before transformation. Therefore, this type of welfare function is independent of changing the scales of the individual utility functions, and inter-user comparisons of utility are not required, which is an interesting property, since *transferability* of utility remains questionable.

*Fairness*, finally, can be seen as an improvement to welfare in that it does not depend on an unspecified welfare function and rather specifies the characteristics of a solution to the allocation problem. An allocation is considered *fair*, if it is *pareto-efficient* and if it has the property of being *equitable*. *Equitable* means the symmetric property that no user wishes to trade his final bundle for another user's final bundle (see [10]). In general, fair allocations will exist and can, for example, be constructed by starting with an equal division allocation and then trading to a pareto-efficient allocation using mechanisms of a competitive market (see, e.g., [11]). Note that the allocated goods need to be freely transferable and divisible; also note that this statement does not necessarily hold for economies with production [15].

## 3 Fairness Concepts in Computer Networks

In the field of computer networks, the term *fairness* corresponds rather to the concept of maximum welfare than to the definition of fairness as given in Section 2. In the following, we will examine different concrete examples and scenarios of the abstract fairness definition given above.

Fairness issues have to be considered both in the end-systems' operating systems and in the network.

Fairness issues concerning the operating system are related to process management, I/O device scheduling, etc., and are not investigated here.

For the network, fairness concepts can be applied to medium access control and to data transport. Concerning fair medium access control (MAC), mechanisms have been investigated for shared physical network links such that on average each sender or receiver gets a fair share of the available bandwidth. This issue is of concern both for medium access control in LAN environments, and, more recently, for mobile networks (see, e.g. [16]) and for all-optical networks (see, e.g., [17]). We will not discuss MAC layer fairness mechanisms in greater detail here, but like to mention that many related problems, such as for instance fair MAC-layer sharing of a common channel under error conditions, are non-trivial and still require further research.

As for data transport, fairness concepts are relevant concerning both elastic traffic and real-time traffic, and for both kinds there exist two approaches for fairness provision: one is to provide fairness by defining appropriate cell/packet scheduling and queue management algorithms on networking nodes, whereas the other one is to achieve fairness by end-to-end flow control mechanisms. In an OSI-like layered view of the network, the former can be regarded as physical layer, data-link layer and network layer mechanisms, whereas the latter are rather transport layer or application layer mechanisms. Introducing multicast cell/packet delivery yields yet another level of complexity concerning fairness, which in all details is still not understood.

Note that fairness issues that have been addressed in current computer networking research mostly concern the problem of fair bandwidth distribution among competing flows. Fair delay management, fair loss rate distribution, and fair jitter control have hardly been addressed at all levels of abstraction, which, in our opinion, is insufficient for future applications with fairness requirements.

In the following, a couple of example fairness definitions for computer networks are presented, the implications of both queue management/scheduling algorithms and end-to-end algorithms for the provision of fairness are given, new challenges concerning multicast communication are identified and some general problems with implementing fairness in computer networks are discussed.

## 3.1 Example Network Fairness Definitions

The following three examples of fairness definition, i.e., *maxmin fairness*, *proportional fairness* and *general weighted fairness* are not intended to cover all imaginable definitions, but we believe that they represent the most common ones for communication networks.

### 3.1.1 Maxmin Fairness

The most popular fairness concept in computer networks is that of *maxmin-fairness*[18]. This fairness definition corresponds to the Rawlsian welfare function $W(u_1, ..., u_n) = min(u_1, ..., u_n)$ with the individualistic utility functions $u_i(x_i) = x_i \forall i \epsilon \{1, ..., n\}$, i.e., for all users $i$, the utility of the resource bundle allocated to user $i$ is represented by the value of the resource bundle itself. Maxmin-fairness yields a solution $x^s = (x_1^s, .., x_n^s)$ for $max(min(x_1, ..., x_n))$. This solution has the property that forall $i$, $x_i^s$ cannot be increased without simultaneously decreasing $x_j^s$ for some $j$ with $x_j^s \leq x_i^s$. For a discussion of *maxmin-fairness*, see, e.g., [19]. Maxmin-fairness in networks gives priority to traffic with a low $x_i$ value. Thus, such traffic is by the definition of maxmin-fairness protected against other more QoS-consuming flows. When, for instance, maxmin-fairness is applied to bandwidth, is does not matter how many networking nodes a traffic flow passes, its fair share remains the same and depends only on the resources available at the bottleneck link/node on the path from source to destination. For that reason, maxmin-fairness is one of the fairness criteria that can be implemented with local information only, which makes it very attractive for many scenarios.

### 3.1.2 Proportional Fairness

Another interesting fairness criterion is *proportional fairness* (see [20])[4]. Using our welfare definitions of Section 2, proportional fairness corresponds to the Nash criterion, i.e., a proportional fair allocation is the solution to the welfare maximization problem with the welfare function $W(u_1, ..., u_n) = \sum_i log(u_i)$ and individualistic utility functions $u_i(x_i) = x_i$. Similarly, *proportional fairness* can be regarded as the solution to the maximization problem of $W(u_1, ..., u_n) = \sum_i u_i$ with $u_i = log(x_i)$. This interpretation is more meaningful when examining fairness for elastic traffic scenarios (see [20]).

The actual definition of proportional fairness is the following: an allocation $x^s$ is *proportionally fair* if it is feasible and if for any other feasible allocation $x^*$, the aggregate of proportional changes is zero or negative, i.e. $\sum_i (x_i^* - x_i^s)/x_i^s \leq 0$. Because the region of feasible allocations is concave and compact, the proportionally fair allocation is the only solution to the maximization of $W(u_1, ..., u_n) = \sum u_i$ with $u_i = log(x_i)$.

This relationship can be seen, if we assume that, for a given solution $x^s = (x_1^s, ..., x_n^s)$ to the maximization problem, there exists an allocation $x^* = (x_1^*, ...x_n^*) = (a_1 * x_1^s, ..., a_n * x_n^s)$ with $a_i$ such that $x_i^* = a_i * x_i^s \forall i$ and for which $\sum_i (x_i^* - x_i^s)/x_i^s > 0$. Thus, it follows that $\sum_i a_i > n$, which is a contradiction to the concavity and compactness of the feasible region of allocations, of which $x^s$ is the maximum.

Proportional fairness does not favour small flows as much as maxmin fairness. For example, when used for bandwidth allocation, proportional fairness punishes flows that pass many bottleneck nodes in terms that they get less bandwidth than in the maxmin case. Thus, proportional fairness better represents the resource usage view: the more resources a flow occupies, the less throughput it perceives. Note that for that reason, proportional fairness cannot be implemented using local mechanisms only. On the other

[4] Frank Kelly provided quite a substantial amount of work on proportional fairness, which can be found at http://www.statslab.cam.ac.uk/˜frank .

hand, since proportional fairness corresponds to the Nash criterion, no inter-personal comparisons of utility are meaningful. This feature has as a consequence that the proportionally fair resource allocation does not change if users perform positive linear transformations of their utility curves. Thus, with proportional fairness, the absolute values of the users utility functions have no meaning

### 3.1.3 General Weighted Fairness

*General weighted fairness*[21] is a fairness scheme of fair local resource allocation and has been defined for ATM ABR service (see [22]). It subsumizes a whole set of fairness criteria as special cases. For a connection $i$, the general weighted fair allocation $g_i$ is defined as $g_i = r_{min_i} + \frac{\omega_i}{\sum_j \omega_j}(bw_{excess} - \sum_i r_{min_i})$ where $bw_{excess}$ is the excess bandwidth to be shared among connections bottlenecked on this link, $\omega_i$ is the preassigned weight associated with the connection $i$, and $r_{min_i}$ is the minimum rate that is required by connection $i$ and that has to be guaranteed before dividing the remaining resources among all connections in a fair manner. With this definition of fairness, different fairness criteria can be achieved by selecting appropriate weights $\omega_i$ and minimum rates $r_{min_i}$.

Note that since *general weighted fairness* is a local scheme, it directly applies only to fairness criteria that require local information only. For fairness criteria that require global knowledge, the weights $\omega_i$ for a connection $i$ might have to be assigned differently on all network nodes traversed by $i$.

For example, if for all connections $i$, $r_{min_i} = 0$ and $\omega_i = c$ with constant $c$, this definition of general weighted fairness yields maxmin-fairness.

## 3.2 Queue Management Mechanisms for Fairness

Providing fairness through queue management and scheduling mechanisms is an approach that has a high impact on the network's architecture, since the fairness algorithms are implemented on switches or routers. But when supported, it can provide the most efficient, flexible and exact mechanism for fairness. Especially, reactions to changes in the network's flow situation can be performed more effectively than in the case when fairness is provided through end-to-end mechanisms only. In addition, fair queue management is needed to protect well-behaved network flows from greedy ones and is necessary for different end-to-end congestion control mechanisms to coexist.

Both for networks that are inherently virtual circuit (like ATM) and for networks that are datagram (like IP), fairness definitions can be implemented by providing an adequate queue management and scheduling algorithm inside the switches or routers.

### 3.2.1 Switched Networks

For example, in the ATM TM 4.0 specification[22], various bandwidth related fairness criteria for the ABR service are defined. *General weighted fairness* (see Section 3.1.3) can be seen as a generalization of these fairness definitions: see [21] for an algorithmic description of the modifications to an existing switch algorithm that achieve general weighted fairness.

### 3.2.2 Routed Networks

For the datagram network case, fairness definitions can be implemented at packet schedulers on routers. For example, node-by-node windowing flow control with round-robin packet scheduling at each node leads to *maxmin-fair* scheduling. Another example of providing fairness mechanisms on routers is to use *fair queueing* algorithms. The mechanism of fair queueing is very simple: each router has multiple input queues (one for each source or source flow) for each output line and scans the input queues, for example, in a round robin fashion for incoming packets. Improved versions of fair queueing perform the round robin scheduling based on bytes instead of packets. There exists a whole range of fair queueing algorithms. For some early examples, see [23] and [24]

The approach of using fair queueing to provide fairness is, for instance, taken by the *user-share differentiation (USD)* scheme[25], which is a proposal for differentiated services[26] that ensures that the bandwidth allocated to traffic from a user is in proportion to the user's share negotiated with the service provider. Implementations of schemes like *USD* use extended versions of fair queueing algorithms like *weighted fair queueing*[24] or variations of it (e.g., *worst-case fair weighted fair queueing*[27], *self-clocked fair queueing*[28], *deficit round robin*[29]).

Although these fair queueing algorithms lead to a more fair bandwidth distribution among competing and not necessarily all well-behaving flows, they have the disadvantage of operating on a per-flow basis, the scalability, robustness and feasiblity of which in high-speed networks are still questionable. This is, because fair queueing algorithms have been designed for congestion control and are usually *stateful*, as opposed to *stateless* congestion control algorithms such as *random early detection (RED)*[30] and its variations.

The discussion of *stateful* versus *stateless* congestion control algorithms can be likened to the QoS architectural design of IntServ's *per-flow-state* model[31] versus DiffServ's stateless flowmodel[26]. Obviously, stateless schemes like *FIFO* or *RED* cannot protect well-behaving flows from misbehaving ones. For this reason, they cannot provide fairness guarantees without appropriate end-to-end control mechanisms.

## 3.3 End-to-end Mechanisms for Fairness

End-to-end mechanisms for fairness implement transport-layer or application-layer algorithms that supply the fairness required.

### 3.3.1 Transport Layer Fairness

Existing end-to-end fairness mechanisms at the transport layer are usually implemented by end-to-end flow control schemes. For instance, additive increase and multiplicative decrease end-to-end flow control, assuming best effort FIFO queueing with tail dropping inside the network, tends to lead to *proportional fairness*[32][33].

The probably most important example of such an additive increase and multiplicative decrease mechanism is TCP. Once an initial threshold is reached, the sending rate is repeatedly controled using the following mechanisms the sending rate receives an additive increase until congestion is detected, upon which the algorithm quickly backs off by reducing the sending rate multiplicatively.

Note that this issue is still not completely understood. For instance, in [34] the issue is revisited and it is found that under certain conditions, proportional fairness can only very appoximately represent the real rate allocations implicitely provided by additive increase and multiplative decrease end-to-end flow control.

The problem with end-to-end fairness mechanisms is that these mechanisms normally only work in a cooperative environment, i.e., if all flows competing for network resources are well-behaved. In the Internet, well-behaved means *tcp-friendly*, which is characterized by the property of behaving similar to a TCP flow through not sending at a higher data rate than a similar tcp flow in the same congestion situation[5]. Still, it is very questionable if tcp-friendliness is a valid assumption in the real world. Besides

[5] For a detailed discussion on tcp-friendly applications and protocols see [35].

TCP traffic, UDP traffic exists in current IP networks and is, for instance, used for real-time flows. The rate control algorithms of UDP applications in practice are not always tcp-friendly and therefore harm the overall fairness. In addition, there exists the risk of malicious TCP implementations that are on purpose not tcp-friendly in order to increase their individual throughput on the cost of regular TCP flows. There are currently no restrictive control mechanisms implemented that punish those flows that are not tcp-friendly, unless this punishment is done by queue management on network nodes as described above, which implicitly influences the type of fairness. One possible approach to cope with this problem is to identify tcp-unfriendly flows at the routers and punish them with appropriate dropping policies. For an interesting discussion of solutions to the tcp-unfriendliness problem see [36].

Another approach is to distinguish between *core* and *edge* of the network. At the edge of the network, admission control and flow policing is performed, which allows receiving (on average) fair allocations even with the core only providing FIFO queueing. An example of such a scheme is presented in [36].

When comparing end-to-end fairness mechanisms to queue management fairness mechanisms, it can be noted that the former normally result in statistical on-average fairness, whereas queueing and scheduling mechanisms allow for a more precice control for fair rate allocations and have a shorter response time to adjust to new network load situations.

### 3.3.2 Application Layer Fairness

At the application layer, there are different levels of abstraction concerning fairness. One level of abstraction also concerns network bandwidth allocation and is highly related to the transport layer issues discussed above: if the network does not provide fairness at the core, adaptive algorithms at the end-systems' applications can lead to fair shares of network traffic. In the Internet, this requirement means that applications should implement an adaptive algorithm that react to network situations in a *tcp-friendly* manner.

As mentioned in Section 1, there exists yet another level of abstraction concerning fairness at the application layer. This flavour of fairness is not as much concerned with fair rate allocations in the network, but rather depends on the semantics of the (fair) applications. In this case, the network is just a (possibly unfair) communication channel and the application implements its own mechanisms to provide fairness. Note that the fairness definitions of Section 2 apply also to this flavour of fairness; just the ressource bundles refer to subjects defined by the semantics of the application instead of network layer metrics such as bandwidth or delay.

A common mechanism for ensuring application-layer fairness is synchronization via transaction control: there exist scenarios where it is important to the user that a certain action is guaranteed to be performed at the same time or during the same time slot as the actions of other users; if not, the system would be considered unfair. For instance, in auction scenarios, synchronization mechanisms are necessary for a fair treatment of the participants: all auctioneers want to get at least the same chance to bet during a certain time slot. In such a scenario, a fair mechanism is to collect (and acknowledge) the bets of all participants in a first step, then to evaluate the synchronized bets and to announce the resulting highest bet as input to the next round. Obviously, such auctioning mechanisms are not very time efficient and scalable for large groups of auctioneers.

For time efficiency, real trading applications do not make use of such synchronization mechanism; the offers are rather processed with price-time priorities: all incoming bets get a time-stamp assigned. At certain intervals all offers that have arrived are examined concerning their offer price and among possibly multiple offers with the highest price the one with the earliest time stamp information is taken (for example, see XETRA[37], the German stock exchange electronic trading system). Considering the fact that the communication channel between the participants of an auction operates in an unfair manner, the fairness achieved with such a scheme has to be questioned.

Note that for all the new application scenarios presented in Section 1, fairness is in general more related to fair timing, which in network terms means that it rather concerns fair delay characteristics than fair bandwidth distribution.

## 3.4 Fairness for Multicast

Concerning network level fairness, multicast packet or cell delivery introduces an additional level of complexity. Following [38], we distinguish between two types of fairness relevant in multicast scenarios: *inter-fairness* and *intra-fairness. Inter-fairness* means that multicast flows should exhibit fair behaviour compared to other, unicast flows. *Intra-fairness* relates to fairness inside the multicast scenario, e.g., different multicast sessions among the same group of senders and receivers should exhibit fairness. Most research related to fairness for multicast concerns inter-fairness. We believe that for intra-fairness in multicast scenarios, there is still a lot of research to be done. In order to further demonstrate the challenges intrinsic to multicast traffic, we briefly discuss two important areas of research in fairness: fairness for *congestion-controlled multicast* and fairness for *layered multicast.*

### 3.4.1 Congestion-controlled Multicast

Similar to the discussion above, for congestion-controlled multicast, switched networks and routed networks can be distinguished. For both scenarios, let us illustrate first examples of fairness provision mechanisms that have been recently presented in the literature. We believe that for both cases, research is still at an early stage and much further investigation is required to fully understand the conceptual and implementation effects of the different methods.

*Switched Networks*

Switched ATM unicast delivery provides several service classes, one of which is the ABR service[22]. ABR is characterized, among other features, by an end-to-end congestion-control paradigm for switched networks: resouce management (RM) cells are sent back from the receiver to the sender to indicate the receivers' experienced quality of service. ABR works in such a way that if the sender adapts its traffic according to this feedback information, low cell loss and a fairly shared bandwidth distribution is experienced[6]. This mechanism can be extended to *intra-fairness* provision for multicast ABR service (see [39] or [40]). The basic idea of the proposed mechanism in [40] is to have RM cells in both directions, i.e., forward and backward RM cells. With the forward cells that are sent downstream along the multicast tree, switches calculate and convey the fair share allocations to their downstream neighbours. With the backward RM cells, information concerning the fair share rate received and operational range and weight[7] of the receivers are transmitted to their up-stream neighbours in the multicast tree. The source, finally, uses this information to compute its appropriate transmission rate by maximizing the global inter-receiver fairness function, based upon the operational ranges, weights and fair share rates of the receivers. For larger groups an aggregated mechanisms that requires less buffering and synchronization at the switches, but allows

[6] The type of fairness obviously depends on the queueing mechanisms implemented.

[7] Jiang et al.[40] assume a *global inter-receiver fairness* function similar to the *weighted-sum-of-utilities welfare* function $W(u_1, ..., u_n) = \sum_i a_i u_i$ presented in Section 2 with weights for the different receivers.

to approximate the inter-receiver fairness function is also presented in [40]. It is clear that this mechanism for fairness in multicast ABR service requires rather substantial changes to the switching algorithms and its scalability still has to be shown.

*Routed Networks*

In routed networks the situation is very similar. There are two classes of congestion control mechanisms: a) using multiple multicast groups for a session with forward error correction (FEC) or layered coding, or b) using one group for each session with rate-based feedback control. Congestion-controlled multicast refers to the latter and means that there exists an end-to-end feedback mechanism indicating to the sender the current congestion situation of the receivers. The objective of the congestion control algorithm is to act according to *inter-fairness* concerning unicast transport layer flows, i.e., in particular, to be TCP-friendly. For a scheme that considers both *inter-fairness* and *intra-fairness* see [41]. We follow the authors of this article in pointing out that is still not entirely clear how inter-fairness among multicast and TCP traffic should be defined: should a multicast session to $n$ receivers get the same share as one TCP connection or as $n$ TCP connections ? We believe that further research should investigate this rudimentary question. A new congestion control scheme for tree-based *reliable multicast* with the property of inter-fairness has recently been presented in [38]. For an overview of research in *reliable multicast*, see, e.g., [42].

*The Loss Path Multiplicity Problem*

A problem with multicast feedback control in general is the *loss path multiplicity (LPM) problem*[43][8], i.e., a packet can be lost on any of the end-to-end paths in the multicast tree. If the sending rate is controled by loss indications from all receivers, there is

[8] also called "problem of the crying baby"

the problem that with an increasing number of such paths the sender will further and further reduce its sending rate until it eventually might cease sending. In [43], it has been shown that with such a scheme of controlling the sending rate, maxmin fair sharing of bandwidth between unicast and multicast traffic is impossible to achieve due to the LPM problem. Also, it has been shown, that if the sender regulates its sending rate according to the most-congested end-to-end path, maxmin inter-fairness can be achieved. Again, it can be questioned whether this approach is the answer to multicast fairness, since the whole traffic flow depends then on possibly one bottlenecked receiver.

### 3.4.2 Layered Multicast

For broadcasting real-time traffic as generated by audio/video applications, layered multicast, is a very interesting mechanism to effectively use network resources in a scenario with heterogenous receivers. Note that there exist various approaches for layered multicast; examples include [44], [45] and [46]. Still, fairness issues for layered multicast have only been investigated at a very basic level, and open a whole new field of future research: for example, in [47], it has recently been shown that for CBR traffic, layered multicast mechanisms that accord to *receiver-driven layered multicast*[44] do not exhibit intra-fairness.

Layered multicast adds another whole new level of complexity to the fairness problem if the different multicast layers operate at different sending rates. A very thorough approach to define and examine multi-rate multicast (inter- and intra-) maxmin fairness has been provided in [48]. Also, a number of interesting properties concerning maxmin fairness in layered multicast scenarios are illustrated: for instance that a mechanism using intra-session coordinated multicast group joins and leaves can achieve multi-rate maxmin fairness.

Even though the first approaches to fairness for layered multicast seem to be very promising, many open issues remain. For instance, it is still unclear, how the fairness mechanisms proposed behave with dynamic sessions, i.e., sessions where receivers join and leave randomly and/or when all receivers are potential senders at the same time. Also, other fairness definitions for layered multicast should be investigated, especially their inter-fairness behaviour concerning congestion-controlled unicast and multicast traffic.

## 3.5 Time-Scale Problems with Fairness

Fairness in general, and in communication networks in particular, has a conceptual problem concerning time-scales. Exact fair QoS allocations can only be calculated for a given moment with a, in that moment, static traffic situation. Obviously, this requirement cannot be fulfilled for fairness criteria that need global information in a dynamically changing heterogeneous network traffic environment, unless for each change in traffic, all influenced paths are re-considered and a new fair allocation is established. For large-scale networks, it is still an open research issue whether exact fairness can be achieved with acceptable overhead. Nevertheless, with both queue management and end-to-end mechanisms for fairness, a certain statistical on-average fairness can be achieved in dynamically changing networks with acceptable effort. With statistical fairness there arises the question of how long the time interval has to be until the network traffic converges to a fair allocation distribution. We believe that this question is very important, but still not entirely investigated. Note that this issue is highly related to the research area of *stability*. We believe that it is very important to understand the relationship between fairness and stability; especially with every new fairness algorithm, stability should be considered as a crucial requirement.

It is clear that the accurateness of fairness mechanisms in general is much higher, if the network provides a homogeneous infrastructure for fairness, i.e., mainly if all switches, routers and end-systems follow the same fairness strategy.

Another important fairness issue concerning time is that of *temporal unfairness.* By *temporal unfairness* we mean that even if exact fairness concerning QoS allocations is provided for competing traffic flows, there exist situations where flows get rejected, because there are no additional resources available, or they get accepted and the already established flows experience a degradation in quality. In the first case, there is a clear preference towards already established traffic flows, i.e., new flows might feel being treated unfairly, whereas in the latter case, everybody has to suffer when the new flow is established, which might cause the whole system to collapse. It is still an open, and probably rather philosophical question, how the problem of temporal unfairness should be tackled. In our opinion, for stability and service predictability purposes, rejection of incoming seems to be more appropriate than degrading already established flows.

## 4 The Fairness Challenge

In this section, the *fairness challenge* with its many facets is introduced. For that reason, the open issues that have been explored earlier in this article are briefly summarized and categorized, and other white spots in the research area of fairness are presented.

The *fairness challenge* in computer networks has two main aspects, the first one of which is *fairness extension* and the second one being *fairness implementation.*

## 4.1 Fairness extension

Extending fairness comprises four types of extensions: *extension of concept*, *extension of definition*, *extension of application*, and *extension of scenario*. We consider all of these extensions to be fruitful for future research in fairness.

### 4.1.1 Extension of concept

Extending the fairness concept mainly concerns the distinction between *welfare* and *fairness*: currently, the fairness examples in computer networks rather represent *welfare* than *fairness*, and it is still an open question if the concept of fairness (as opposed to welfare) as introduced in Section 2 can be incorporated into computer networks. With pricing and trading mechanisms in networks, the fair allocation could be reached through trading to a pareto-efficient optimum. Possibly, other mechanisms to ensure that an allocation is *equitable* exist and could be implemented in a different manner. In addition, the extension of concept encompasses considering the applicability of other fairness criteria to computer networks.

### 4.1.2 Extension of definition

Currently, fairness is mainly defined for unicast cell or packet delivery. Other types of delivery, such as *multicast*, *broadcast* or *anycast*[49] require an extension of the fairness definition. Examples for multicast include the aspects of *inter-fairness* and *intra-fairness* for best-effort multicast, congestion-controlled multicast, reliable multicast, and layered multi-rate multicast. Also, solutions to the *LPM* problem for different fairness criteria and multicast scenarios with multiple senders and dynamically joining and leaving receivers are at an early stage and worthwhile further investigation. In all of these topics, research has just begun.

### 4.1.3 Extension to other QoS parameters

The *extension to other QoS parameters* means to not only apply fairness concepts to bandwidth distribution problems, but also consider the fair management and control of *loss rate*, *delay* and *delay jitter*. We believe that especially fair delay management and fair jitter control have to be considered for future applications that require fairness as part of service quality.

### 4.1.4 Extension to new applications

*Extension to new applications* also deals with the aspect of service quality: the current fairness concept in computer networks has to be extended up to the application level, i.e., fair[9] applications should be supported. We believe that such application-semantic fairness is best supported if the application's communication channel provides the necessary degree of fairness. An integral and comprehensive approach for fairness provisioning is needed and available as a new field for future research in fairness.

### 4.1.5 Extension of scenario

With new networking technologies, new challenges for consideration and incorporation of fairness arise. Such new networking technologies comprise mobile communication, radio or satellite communication, all optical networks, and many many more. Each of these technologies produces a whole number of new scenarios, in which fairness issues become relevant and should be investigated.

## 4.2 Fairness implementation

Implementing a certain type of fairness in a computer network entails many challenges. We like to mention some of the more important ones that have to be considered: the

[9] as defined by the semantics of the application

first problems when incorporating fairness, especially statistical fairness is the *time-scale problem* discussed in Section 3.

Another challenge to be considered is the cooperation problem for end-to-end fairness mechanisms: how can fairness be achieved via end-to-end only mechanisms if there are malicious and greedy flows, in particular if not all flows are *tcp-friendly* ?

Similarly, fairness mechanisms should also focus on real-time applications and allow more fairness criteria than simple tcp-like fairness. Note that for these applications, delay is more important than throughput, and therefore, mechanisms for fair delay distribution should be explored.

*Priority management* is another issue that can be directly linked to fairness. Flows with different priorities should be treated differently in terms of the fairness they perceive; for instance, high priority flows might always be treated in a maxmin fashion, while low priority flows are managed using a different scheme, or, in situations with *temporal unfairness*, higher priority flows could be assigned precedence. We believe that a clear understanding of how priority management and fairness can be combined would lead to a better integral concept of fairness in computer networks.

Another important area of research in fairness is concerned with *fairness for aggregated flows*: modern networking technologies distinguish between the core network and the access network. The application wants to perceive per-flow fairness, which should be represented via appropriate access network mechanisms. Nevertheless, in the core network, the requirement for high-speed transmission and scalability leads to aggregated stateless traffic forwarding. For that reason, some mapping of per-flow fairness in the access network to aggregated-flow fairness in the core-network is needed in order to assure end-to-end fairness in such environments. Some first approaches are being developed, but there is lots of room for innovative concepts to cope with that problem.

## 5 Conclusion

In this paper, the issue of *fairness* in computer networks has been investigated. The similarities and linkage between fairness concepts of political economics and computer networks have been presented and existing research in network fairness at various levels of abstraction ilustrated. In order to further motivate research in fairness, some application scenarios have been briefly described that require fairness issues as an important part of their application-specific service quality.

With this state-of-the-art overview of research in fairness, we have pointed out many aspects and open questions available for future research. We hope to have demonstrated that even though specific fairness issues concerning computer networks have already been investigated to some extent, there is a vast amount of interesting and challenging work left to be done.

For a first classification of fairness issues to be examined, we presented the *fairness challenge* and hope to have motivated the reader to participate in further investigation of the wide range of interesting and challenging open topics in the field of fairness in computer networks.

## References

[1] A. Maslow, *Motivation and Personality*, Harper & Row, New York, 1954.

[2] D. K. Foley, "Resource allocation in the public sector," *Yale Econ. Essays*, vol. 7, pp. 73–76, 1967.

[3] Elisha A. Pazner, "Pitfalls in the Theory of Fairness," *Journal of Economic Theory*, vol. 14, pp. 458–466, 1977.

[4] H. Steinhaus, "Sur la division pragmatique," *Econometrica (supplement)*, vol. 17, pp. 315–319, 1949.

[5] L. E. Dubins and E. H. Spanier, "How to cut a cake fairly," *American Mathematical Monthly*, pp. 1–17, 1961.

[6] Harold W. Kuhn, "On Games of Fair Division," in *Essays in Mathematical Economics*, Martin Shubik, Ed. 1967, pp. 29–37, Princeton University Press.

[7] Hal R. Varian, *Intermediate Microeconomics - A Modern Approach*, W.W. North & Company, New York/London, fifth edition, 1999.

[8] Scott Shenker, "Fundamental Design Issues for the Future Internet," *IEEE Journal Selected Areas Communication*, vol. 13, pp. 1176–1188, 1995.

[9] K.J. Arrow and F.H. Hahn, *General Competitive Analysis*, Oliver and Boyd, Edinburgh, 1971.

[10] Hal R. Varian, "Distributive Justice, Welfare Economics, and the Theory of Fairness," *Philosophy & Public Affairs*, vol. 4, no. 3, pp. 223–247, 1975.

[11] Hal R. Varian, *Microeconomic Analysis*, Norton, New York, third edition, 1992.

[12] Kenneth J. Arrow, *Social Choice and Individual Values*, Wiley, New York, 1963.

[13] Donald Wittman, "A Diagrammatic Exposition of Justice," *Theory and Decision*, vol. 11, pp. 207–237, 1979.

[14] J.F. Nash, "The Bargaining Problem," *Econometrica*, vol. 18, pp. 155–162, 1950.

[15] E. Pazner and D. Schmeidler, "A difficulty in the concept of fairness," *Rev. Econ. Studies*, vol. 41, pp. 441–443, 1974.

[16] J. R. Moorman and J. W. Lockwood, "Multiclass Priority Fair Queueing for Hybrid Wired/Wireless Quality of Service Support," in *Proceedings of WoWMoM99*, Seattle, WA, Aug. 1999.

[17] M. A. Marsan, A. Bianco, E. Leonardi, A. Morabito, and F. Neri, "All-Optical WDM Multi-Rings with Differentiated QoS," *IEEE Communications Magazine*, pp. 58–66, Feb. 1999.

[18] J. Jaffe, "Bottleneck flow control," *IEEE Transactions on Communications*, vol. 7, no. 29, pp. 954–962, July 1980.

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

[20] Frank P. Kelly, "Charging and rate control for elastic traffic," *European Transactions on Telecommunications*, vol. 8, pp. 33–37, 1997.

[21] Bobby Vandalore, Sonia Fahmy, Raj Jain, Rohit Goyal, and Mukul Goyal, "General Weighted, Fairness and its Support in Explicit Rate Switch Algorithms," *submitted to Computer Communications journal*, February 1999.

[22] Shirish S. Sathaye, "ATM Forum Traffic Management Specification Version 4.0," *ftp://ftp.atmforum.com/pub/approved-specs/af-tm-0056.0000.pdf*, 1996.

[23] A. Demers, S. Keshav, and S. Shenker, "Analysis and Simulation of a Fair Queueing Algorithm," in *Proceedings of ACM SIGCOMM'89*, 1989, pp. 3–12.

[24] A. K. Parekh and R. G. Gallager, "A Generalized Processor Sharing Approach to Flow Control - the Single Node Case," in *Proceedings of IEEE INFOCOM'92*, May 1992.

[25] Z. Wang, "User-Share Differentiation - Scalable Service Allocation for the Internet," *Internet Draft*, Nov. 1997.

[26] Y. Bernet et. al., "A framework for differentiated services," Internet Draft, draft-ietf-diffserv-framework-01.txt, November 1998.

[27] J.C.R. Bennett and H. Zhang, "$WF^2Q$: Worst-case Fair Weighted Fair Queueing," in *Proceedings of IEEE INFOCOM'96*, March 1996, pp. 120–128.

[28] S. Golestani, "A Self-clocked Fair Queueing Scheme for Broadband Applications," in *Proceedings of IEEE INFOCOM'94*, April 1994, pp. 636–646.

[29] M. Shreedhar and G. Varghese, "Efficient Fair Queueing using Deficit Round Robin," in *Proceedings of ACM SIGCOMM'95*, September 1995, pp. 231–243.

[30] S. Floyd and V. Jacobson, "Random early detection for congestion avoidance," *IEEE/ACM Transactions on Networking*, vol. 1, no. 4, pp. 397–413, July 1993.

[31] S. Shenker, R. Braden, and D. Clark, "Integrated services in the Intenet architecture: an overview," Internet RFC 1633, June 1994.

[32] Frank P. Kelly, A.K. Maulloo, and D.K.H. Tan, "Rate control for communication networks: shadow prices, proportional fairness and stability," *Journal of the Operational Research Society*, vol. 49, 1998.

[33] L. Massoulié and J. Roberts, "Fairness and quality of service for elastic traffic," *CNET-France Tél'ecom*, Feb. 1998.

[34] Paul Hurley, Jean-Yves Le Boudec, and Patrick Thiran, "A Note on the Fairness of Additive Increase and Multiplicative Decrease," **???**, Oct. 1998.

[35] J. Mahdavi, "The TCP-friendly web site," http://www.psc.edu/networking/tcp_friendly.html.

[36] Ion Stoica, Scott Shenker, and Hui Zhang, "Core-Stateless Fair Queueing: Achieving Approximately Fair Bandwidth Allocations in High Speed Networks," *ACM SIGCOMM'98*, pp. 118–130, Oct. 1998.

[37] "Xetra release 3," http://www.xetra.de.

[38] I. Rhee, N. Balaguru, and G. Rouskas, "MTCP: Scalable TCP-like Congestion Control for Reliable Multicast," in *Proceedings of IEEE INFOCOM'99*, New York, NY, Mar. 1999.

[39] H. Tzeng and K. 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.

[40] T. Jiang, M. Ammar, and E. Zegura, "Inter-Receiver Fairness: A Novel Performance Measure for Multicast ABR Sessions," in *Proceedings of ACM SIGMETRICS'98*, Madison, Wisconsin, June 1998.

[41] H. A. Wang and M. Schwartz, "Achieving Bounded Fairness for Multicast and TCP Traffic in the Internet," in *Proceedings of ACM SIGCOMM'98*, Vancouver, Canada, Sept. 1998.

[42] B. N. Levine and J.J. Garcia-Luna-Aceves, "A comparison of known classes of reliable multicast protocols," in *Proceedings of ICNP*, Oct. 1996.

[43] S. Bhattacharyya, D. Towsley, and J. Kurose, "The Loss Path Multiplicity Problem for Multicast Congestion Control," in *Proceedings of IEEE INFOCOM'99*, New York, NY, Mar. 1999.

[44] S. McCanne and V. Jacobson, "Receiver-Driven Layered Multicast," in *Proceedings of ACM SIGCOMM'96*, Oct. 1996.

[45] A. Banchs, W. Effelsberg, Ch. Tschudin, and V. Turau, "Active Multicasting of Multimedia Streams," in *Proceedings of the IEEE Local Computer Networks Conference LCN'98*, October 1998, pp. 150–159.

[46] L. Vicisano, L. Rizzo, and J. Crowcroft, "TCP-like congestion control for layered multicast data transfer," in *Proceedings of INFOCOM'98*, Mar. 1998, pp. 996–1003.

[47] R. Gopalakrishnan, J. Friffioen, G. Hjálmtýsson, and C.J. Sreenan, "Stability and Fairness Issues in Layered Multicast," in *Proceedings of ???*, Mar. 1999.

[48] D. Rubenstein, Jim Kurose, and Don Towsley, "The Impact of Multicast Layering on Network Fairness," in *Proceedings of ACM SIGCOMM'99*, Sept. 1999.

[49] Christian Huitema, *IPv6: The New Internet Protocol*, Prentice Hall, Englewood Cliffs, New Jersey, 1996.