# Effairness: A Metric for Congestion Control Evaluation in Dynamic Networks

Sergey Gorinsky Harrick Vin

Technical Report TR2001-31
Laboratory for Advanced Systems Research
Department of Computer Sciences
The University of Texas at Austin
Taylor Hall 2.124, Austin, TX 78712, USA
{*gorinsky,vin*}*@cs.utexas.edu*

August 2001

*Abstract*— **This paper examines the problem of congestion control evaluation in dynamic networks. We determine a source of deficiencies for existing metrics of congestion control performance – the existing metrics are defined with respect to ideal allocations that do not represent short-term efficiency and fairness of network usage in dynamic environments. We introduce the concept of an *effair allocation*, a dynamic ideal allocation that specifies optimal efficiency and fairness at every timescale. This concept has a general applicability; in particular, it applies to networks that provide both unicast and multicast services. Another desirable property of the effair allocation is its dependence on the communication needs and capabilities of applications. We design an algorithm that accounts for network delays and computes the effair allocation as a series of static ideal allocations. Using the notion of effair allocation as a foundation, we define a new metric of *effairness* that shows how closely the actual delivery times match the delivery times under the effair allocation.**

## 1 Introduction

Efficient and fair allocation of network resources is a primary objective in congestion control. Many network applications – such as Web browsing or distributed multimedia – are interested in short-term fairness and efficiency of their data delivery. For example, what matters for a video streaming application is not only its throughput averaged over the stream duration but also when the receiver obtains individual frames. In this paper, we argue that existing metrics of congestion control performance are poorly suitable for representing *short-term* efficiency and fairness under *dynamic* network conditions.

We observe that any metric of congestion control performance is defined with respect to an *ideal allocation* which exhibits optimal efficiency and fairness. The deficiency of the existing metrics results from their inappropriate choice of the underlying ideal allocation: these metrics are defined with respect to *static* or *long-term* ideal allocations. To be able to represent short-term efficiency and fairness in dynamic networks, a metric should be defined with respect to an ideal allocation that specifies optimal efficiency and fairness at any timescale.

Below, we discuss the properties a metric and its underlying ideal allocation should possess to be useful for evaluation of congestion control in dynamic networks:

- *Representativity* is an obvious and, at the same time, the most important feature for a metric of congestion control performance. The values of the metric should provide applications with *dependable* and *meaningful* information about the actual efficiency and fairness.

  A related consideration is the subject of measurement. The design of a congestion control mechanism is not a goal by itself. The goal is a fair and efficient allocation of network resources. Thus, the metric should represent the *actual allocation* rather than changes in the internal state of the evaluated mechanism (such as changes in the congestion window or sending rate of an end-point congestion control protocol).

- The underlying ideal allocation of a metric should have a *simple specification* in order to facilitate analysis of congestion control designs.

- If an application receives a smaller share of a bottleneck resource than other applications, it is impossible to determine whether the resource allocation is fair or unfair without knowing the demand of this application. Similarly, an underutilized network does not necessarily mean inefficiency of its congestion control mechanisms – the lack of demand can be a reason for the underutilization. Thus, a metric and its underlying ideal allocation should depend on *communication needs and capabilities of applications*.

- The Internet has not only diverse traffic types but also various means for congestion control including load adjustment at end-points, active queue management in the network, and multicast routing (which allows a multicast receiver to control congestion by selecting an appropriate subscription level in a layered multicast session). Therefore, the metric should be *generally applicable* in order to present efficiency and fairness of congestion control designs in the perspectives of unicast and multicast sessions, long-lived and short-lived sessions, file transfers and those multimedia applications that are interested in performance on small timescales.

  While evaluating short-term efficiency and fairness of resource allocation, one cannot ignore the propagation delays from the shared resources to the end-points of applications. The differences between these delays can be significant on the timescale of such evaluation. Consequently, the definition of an ideal allocation should explicitly consider the *distributed nature* of networks and account for *delay characteristics* of network paths.

In this paper, we propose an ideal *effair allocation* and a metric of *effairness* that satisfy the stated requirements. The effair allocation is a dynamic ideal allocation that specifies optimal efficiency and fairness at every timescale. Assuming the fluid traffic model, the effair allocation defines a dynamic rate for each receiver in the network. This definition applies, for instance, to networks with multicast routing. The introduced metric of effairness shows how closely the actual delivery times match the delivery times under the effair allocation.

The rest of the paper is structured as follows. Section 2 examines existing approaches to evaluation of congestion control mechanisms. Section 3 presents our metric of effairness and designs an algorithm for computing the effair allocation. Section 4 illustrates the usefulness of our metric. Finally, Section 5 summarizes the contributions made by this paper.

## 2 Related Work

As we mention above, an ideal allocation serves as a foundation for evaluating any actual resource allocation. In Section 2.1, we consider two traditional types of ideal allocations. While Section 2.2 describes how the existing approaches evaluate congestion control designs with respect to these ideal allocations, Section 2.3 discusses limitations of the existing approaches.

### 2.1 Ideal Allocations

The existing studies of congestion control use two types of ideal allocations: *static* and *long-term*.

*Static* approaches formulate the congestion control problem based on the abstraction of a flow [4, 10]. A flow forms a path of network links from the sender of an application to its receiver. When flows share links, a global principle of fair sharing – such as maxmin fairness [4, 10] or proportional fairness [13, 14] – determines the distribution of the link capacities between the flows. The rate of a flow is defined as a number representing the capacity allocated to the corresponding application. The *static ideal allocation* is such an assignment of rates to flows that conforms to the selected principle of fair sharing. The lack of consensus on a single principle of fair sharing has resulted in a multiplicity of static ideal allocations, e.g., the *maxmin-fair* and *proportionally-fair* allocations.

Note that static ideal allocations contain no reference to time. As Section 2.3 shows, this feature makes them poorly suitable as a basis for evaluation of congestion control designs in dynamic environments.

The most dominant among *long-term* approaches to characterizing an ideal allocation is TCP-*friendliness* which has emerged from practical concerns about coexistence of traditional TCP sessions with new traffic types in the Internet [15, 18, 19]. A common definition of TCP-friendliness is based on the approximate statistical relationship between the long-term throughput of a TCP session and other parameters such as the round-trip time and frequency of loss indications. According to this equation-based definition, congestion control is TCP-friendly if it provides statistically the same long-term throughput as TCP for the same round-trip time and frequency of loss indications.

While a static ideal allocation is global (it defines rates for all the flows in the network), a TCP-friendly ideal allocation is local: it specifies an ideal throughput only for a particular unicast application. Another important difference is that TCP-friendliness includes a notion of time – the ideal throughput is specified with respect to some long period.

In the next section, we review metrics used for evaluation of congestion control designs with respect to the described ideal allocations.

### 2.2 Traditional metrics

First, let us examine traditional evaluation methodologies when a static ideal allocation (e.g., the maxmin-fair allocation) serves as a basis for assessing a congestion control design. These methodologies put a major stress on verification whether or how closely the actual throughputs of applications converge under static network conditions to the rates specified by the ideal static allocation. To evaluate performance of a congestion control design in dynamic environments, the methodologies employ a metric of *convergence time* to characterize how quickly the actual allocation approaches the static ideal allocation after introduced disturbances such as the start or termination of a session [21].

Evaluation of congestion control algorithms with respect to long-term ideal allocations is similar. It attributes a primary importance to examining whether or how closely the actual

throughput of the controlled application matches – over long intervals – the TCP-friendly ideal throughput. Since TCP-friendliness specifies a local allocation for a particular application, this examination can be conducted in a dynamic environment where other sessions join and leave the network. The *throughput ratio* is a metric that quantifies the degree of the closeness between the actual long-term throughput of the application and its TCP-friendly ideal throughput [2, 15, 19]. Since the throughput ratio represents only the long-term allocation of resources, researchers employ additional metrics such as *aggressiveness*, *smoothness*, *responsiveness*, *stabilization time* to assess short-term efficiency and fairness of the congestion control design [3, 8]. These additional metrics characterize how quickly the congestion control algorithm reacts to changes in network conditions.

## 2.3 Limitations

Below, we analyze limitations of the proposed metrics in regard to the desired properties from Section 1.

### 2.3.1 Representativity

Let us consider the metric of *convergence time* used in the methodologies that are based on the static ideal allocation. In dynamic networks, the session population keeps changing. When the frequency of the changes is high, there can be a persistent mismatch between the actual allocation and the static ideal allocation that corresponds to the current population of sessions in the network. Since the time interval between subsequent changes can be consistently insufficient for the congestion control algorithm to converge to the new static ideal allocation, the metric of convergence time does not provide a meaningful representation of how efficient and fair the algorithm is in dynamic environments. This inadequate representativity of convergence time results primarily from the choice of a static ideal allocation as a basis for defining the metric. Thus, to represent efficiency and fairness in dynamic networks, a metric of congestion control performance should be defined with respect to an ideal allocation that contains a notion of time.

The TCP-friendly metric of the *throughput ratio* fares better in a dynamic environment: it reliably represents efficiency and fairness of the long-term allocation for a particular application. On the other hand, this metric does not reflect short-term efficiency and fairness. Consequently, measuring the throughput ratio does not yield meaningful information for short-lived sessions or those applications that are interested in maintaining fair and efficient delivery even over short periods during their long presence in the network.

The desire to assess performance of TCP-friendly congestion control algorithms on small timescales has led to additional metrics such as *aggressiveness*, *smoothness*, *responsiveness*, and *stabilization time*. Unfortunately, these metrics have a very limited value for assessing an actual short-term allocation because they merely measure how promptly the congestion control algorithm adjusts the load of the controlled application on the network. The information whether the algorithm changes the sending rate of the application by 1 Kbps or 5 Kbps over a round-trip time provides little indication of how fair and efficient the resulting allocation is. The poor representativity of these additional metrics derives from the fact that TCP-friendliness is a long-term approach and does not specify a short-term ideal allocation. Without devising such ideal allocation, it seems impossible to define a metric that represents efficiency and fairness of actual short-term allocations.

### 2.3.2 Specification of the ideal allocation

To facilitate analysis of congestion control designs, it is preferable if the underlying ideal allocation of a metric has a simple specification. Static ideal allocations usually satisfy this condition. For example, the maxmin-fair allocation has the following simple specification: the rate of any flow cannot be increased without decreasing the rate of another flow to an even lower value. On the other hand, the equation-based specification of the TCP-friendly ideal allocation is quite complex: it contains many parameters such as the rate of loss indications, round-trip time, packet size, and the setting of the TCP retransmission timer. This complexity makes it difficult to conduct theoretical reasoning about congestion control designs with respect to their TCP-friendliness.

### 2.3.3 Dependence on application demands

Although a desirable methodology for evaluating congestion control performance should take into account the communication needs and capabilities of applications (see Section 1), all the examined metrics ignore these factors and assume that applications can utilize as much bandwidth as the network can provide. For example, the equation-based specification for the TCP-friendly ideal allocation is derived under the assumption that the controlled application always has data for transmission. There exist numerous applications that do not fit this assumption: multimedia applications where the useful rate of data delivery has an upper limit, or applications that transmit sequences of bursts separated by periods of low-rate transmission. The throughput ratio and alike metrics of efficiency and fairness fail to account for the demands of such applications.

### 2.3.4 General applicability

As Section 2.3.1 shows, neither of the discussed metrics represents efficiency and fairness of short-term allocations adequately. Thus, one cannot apply these metrics to assess allocations given to short-lived sessions. This feature undermines the applicability of the existing metrics dramatically since short-lived sessions constitute an evident majority in the Internet.

Let us now consider applicability of the metrics in multicast-capable networks. Although all the metrics have originated in the context of unicast communication, there exist multicast extensions to their definitions.

Due to the simplicity of specification for static ideal allocations, multicast versions of these allocations can be easily defined [21, 22]. For instance, one can specify a multicast maxmin-fair allocation in terms of receiver rates (not the rates of flows as in the unicast maxmin-fair allocation) with the convention that the rate of a multicast session on a link is the maximum of the rates among those receivers of the session that are located behind this link [21]. However, such multicast extensions to a static ideal allocation are also static. If a metric is defined with respect to them, it suffers from the same problems (e.g., poor representativity in dynamic environments) as its unicast equivalents.

Multicast extension for the concept of TCP-friendliness seems to be a larger challenge. In particular, one problematic feature of the TCP-friendly unicast allocation is its dependence on the round-trip time. A number of studies argue that to be applicable to layered multicast sessions, the definition of a TCP-friendly multicast allocation should eliminate such dependence [5, 24].

A fundamental obstacle for general applicability of TCP-friendliness is its definition of an ideal allocation in terms of the behavior exhibited by TCP, a specific unicast protocol for end-point congestion control. The resulting linkage to TCP specifics – such as designation for unicast, dependence on round-trip times, congestion detection through loss inference, load adjustment at end-points – makes it difficult to adapt TCP-friendliness and its metrics for congestion control evaluation in general network infrastructures that support multicast routing, different methods of congestion detection (e.g., explicit rate notification), and load adjustment inside the network (e.g., active queue management).

## 3 Definition of effairness

In this section, we define a metric of *effairness* which addresses the above limitations of the existing metrics.

As Section 2.3.1 reveals, the poor representativity of the examined metrics results from inappropriate choices of their underlying ideal allocations (static or long-term). To make a metric representative in dynamic environments, the underlying ideal allocation should be *dynamic* – it should be defined for any timescale, even for short time intervals. We introduce such a dynamic ideal allocation and refer to it as an *effair allocation*.

To specify the effair allocation, we first determine the periods when the ideal allocation does not change; the effair allocation is then defined as a series of these static ideal allocations. Note that since we define the effair allocation based on a static ideal allocation, a different choice of the underlying static allocation yields a different type of the effair allocation. For example, if the underlying static allocation is maxmin-fair, then the result is the *maxmin effair allocation*; the choice of the proportional-fair static allocation leads to the *proportional effair allocation*.

An advantage of this approach is that the specification of the dynamic ideal allocation inherits the *simplicity* characteristic for the specifications of static ideal allocations. Also due to this reliance on static ideal allocations, the definition of the effair allocation can easily be made applicable in *multicast-capable networks*. However, determination of the periods when the ideal allocation is invariable presents a major challenge. To resolve this problem, one should take into consideration the *distributed nature* of networks: because the delays between different application end-points and a shared network link can be different, a change in the allocation for one application can impact the allocation for another application not instantaneously but with a shift in time.

Our methodology for computing the effair allocation is based on a notion of a *stream* that accounts for:

- the impact of *network delays*,
- possible presence of *multicast sessions*, and
- *demands and capabilities of applications*.

Before we give a formal definition for a stream in Section 3.2 and present the computation methodology in detail, we explain how our metric of effairness is defined with respect to the computed effair allocation.

To be meaningful for network applications, our metric tries to capture the most important aspect for an application in the actual allocation of resources: *how long* does it take to deliver *needed data* to the receiving end-points of the application? In the context of some applications such as multimedia, these *needed data* refer not to the total delivered amount but to smaller application-specific chunks, e.g., video frames. To represent this information, we measure three parameters $a_k$, $s_k$, and $f_k$ of *actual* performance for each receiver $k$ where $a_k$ is the *amount of data* delivered to receiver $k$, *start time* $s_k$ is the time when the sender started transmitting these data to receiver $k$, and *finish time* $f_k$ is the time when receiver $k$ received all these data.

Our metric of effairness specifies how close the actual delivery time $f_k - s_k$ is to the *effair delivery time* $\phi_k$ which is the amount of time it would take to deliver the data amount $a_k$ to receiver $k$ under the effair allocation:

**Definition 3.1** *Effairness $e_k$ for receiver $k$ is:*

$$e_k = \frac{\min\{f_k - s_k, \phi_k\}}{\max\{f_k - s_k, \phi_k\}} \quad (1)$$

*where $s_k$, $f_k$, and $\phi_k$ are respectively the start time, finish time, and effair delivery time for receiver $k$.*

The introduced index of effairness takes its values from the range between 0 and 1 where the value of 1 represents the totally efficient and fair allocation.

Since some applications can be represented by multiple streams with multiple receivers (see Section 3.2 for details), we also define effairness for an application – as the average of the effairness indexes among its receivers:

**Definition 3.2** *Effairness $e_\alpha$ for application $\alpha$ is:*

$$e_\alpha = \frac{1}{K} \sum_{k \in \alpha} e_k \tag{2}$$

*where $K$ is the number of receivers $k$ that represent application $\alpha$.*

We similarly define effairness $E$ of the whole network allocation in terms of the effairness indexes for its applications:

**Definition 3.3** *Effairness $E$ of the overall network usage is:*

$$E = \frac{1}{N} \sum_{\alpha} e_\alpha \tag{3}$$

*where $N$ is the number of applications $\alpha$ in the network.*

Such a hierarchical definition of effairness allows us to evaluate congestion control designs both from the application and network perspectives.

Below, we consider how to compute effairness. Section 3.1 states our assumptions. We formally define a *stream* – a primary entity in our approach to congestion control evaluation – in Section 3.2. To represent interaction of streams, Section 3.3 introduces a notion of an *effairness graph* and then presents an algorithm for computing effairness in practical network configurations. Throughout the rest of Section 3, we give a series of examples to illustrate the introduced notions as well as the presented algorithm.

## 3.1 Assumptions

We define the effair allocation in terms of rates and thereby assume fluid traffic, i.e., traffic that consists of infinitesimal particles. Consequently, our model does not represent packet transmission delays. This is a conscious choice. While the packet-switching technology is currently dominant, it is not the only option in the design space for data communication. Networks can deliver data without splitting it into packets that carry control information in their headers. Thus, we exclude the notions of a packet and its header (which are specifics of particular network implementations) from our definition of an ideal allocation and consider only application data, communication of which is the fundamental goal of any network.

Also, we ignore queuing delays in routers – in the ideal allocation, senders transmit data so that routers forward it without delay. Hence, delays in our model for calculating the effair allocation include only the inevitable propagation delays.

Definition 3.1 of effairness refers to start time $s_k$ at the sender and finish time $f_k$ at the receiver. Thus, precise computation of effairness assumes clock synchronization between senders and receivers. We would like to emphasize that the issue of distributed clocks is inherent to evaluating the efficiency and fairness of data delivery between distributed entities. There are two reasons why this issue does not undermine the usefulness of our approach. First, a vast majority of studies evaluate congestion control designs by centralized simulation tools such as ns-2 [17]; these evaluations do not face the problem of clock synchronization. Second, assessment of congestion control in real networks is usually conducted in an environment where the evaluators have control over the end-points and thus can take steps for resolving the problem of distributed clocks.

Finally, we assume that data follows some known path through the network. This is a standard assumption in congestion control research.

## 3.2 Streams

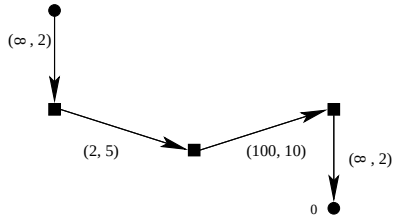
We formulate the congestion control problem in terms of *streams*. A stream is such a generalization of the traditional concept of a flow [4, 10] that can be used to represent multicast sessions, delay characteristics of network paths, and communication needs of applications.

To obtain an abstraction applicable to *multicast sessions*, we define a stream as a tree of directed edges. The leafs of the tree represent end-points of a distributed application and are denoted graphically as circles (see Figure 1). Depicted as squares, the internal nodes of the tree represent network hosts and routers. The direction of edges shows the direction of data delivery from the only sender of the stream to potentially multiple receivers.

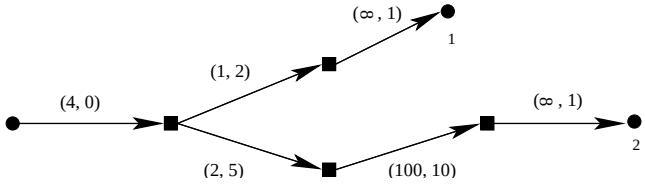
Streams – unlike flows – reflect *delay characteristics* of network paths. An ordered two-tuple ($c$, $d$) annotates each edge where $c$ and $d$ are respectively referred to as the *capacity* and *delay* of the edge. An edge between internal nodes of the tree represents a network link between corresponding hosts or routers; $c$ and $d$ in the ($c$, $d$) annotation of this edge denote respectively the total capacity and propagation delay of the link. Note that this annotation depicts the maximal performance of the link, and not the fair and efficient allocation of the link capacity to an application.

The *communication needs and capabilities of applications* are represented in streams by edges that are incident with the leafs of the tree. In the ($c$, $d$) annotation of such an edge, $c$ represents the bandwidth that the corresponding end-point (the sender or a receiver) needs and is capable to utilize while $d$ specifies the delay to transfer data between the end-point and the network.

We believe that the concept of a stream provides a sim-




(a) unicast session




(b) multicast session

**Figure 1**: Examples of streams.

ple yet comprehensive representation for the application demands and capabilities as well as for the network path characteristics of various distributed applications. For instance, streams with one receiver denote unicast delivery. A unicast or multicast session that performs a file transfer is represented by a single stream. Since the same stream can have receivers $k$ with different start times $s_k$, one stream can represent a multicast session where some receivers join late. Applications with multiple senders are represented using multiple streams. An application that is interested in short-term efficiency and fairness (such as a multimedia application) can also benefit from a multiple-stream representation: this application can split all its delivered data into smaller chunks and use a separate stream for representing the delivery of each chunk.

The following example shows how network applications can be represented as streams.

**Example 1** *Consider a file transfer application where the sender starts its transmission at time $s_0 = 100$ ms and the receiver obtains $a_0 = 900$ Kb of data by time $f_0 = 725$ ms. The stream in Figure 1a denotes this unicast session. The session traverses two network links. The first link has the bandwidth of 2 Mbps and the propagation delay of 5 ms; the bandwidth and propagation delay for the second link are 100 Mbps and 10 ms respectively. The annotations (2, 5) and (100, 10) for the corresponding edges of the stream represent these bandwidths and delays. The annotations ($\infty$, 2) for the edges incident with the leafs of the stream mean that the session can use as much capacity as the network can provide and that the session transfers its data to the network and from the network with the delay of 2 ms.*

*The stream in Figure 1b represents a multicast session with two receivers. The sender starts transmitting $a_1 = a_2 = 500$ Kb of data at time $s_1 = s_2 = 0$. The top multicast receiver $1$ and the bottom multicast receiver $2$ obtain these data by finish times $f_1 = 526$ ms and $f_2 = 541$ ms respectively. The session traverses three network links. The annotations (1, 2), (2, 5), (100, 10) for the corresponding edges of the stream mean that the bandwidth and propagation delay for these links are respectively 1 Mbps and 2 ms, 2 Mbps and 5 ms, and 100 Mbps and 10 ms. The sender can sustain a transmission rate up to 4 Mbps and transfers data into the network without delay. This information is reflected in the annotation (4, 0) for the edge incident with the sender. The annotations ($\infty$, 1) for the edges incident with the receivers denote that the receivers can use as much bandwidth as the network can provide and that each receiver transfers its data from the network with the delay of 1 ms.* ■

To characterize the dynamic allocation of the network capacity to a stream, we specify an *effair rate* $r_k(t)$ at time $t$ for each receiver $k$ of the stream. In conjunction with the edge delays of the stream, the effair rates of the receivers completely define the effair allocation for this stream. For every edge $n$ of stream $\sigma$, the effair rate $r_{\sigma,n}(t)$ of stream $\sigma$ on edge $n$ is formally defined as:

$$r_{\sigma,n}(t) = \max_{l} \; r_l(t + d_{n,l}) \tag{4}$$

where $l$ refers to such a receiver of stream $\sigma$ that lies behind edge $n$ while $d_{n,l}$ denotes the propagation delay from edge $n$ to receiver $l$. In particular, the effair rate $r_\sigma(t)$ of stream $\sigma$ at its sender equals:

$$r_\sigma(t) = \max_{k} \; r_k(t + d_k) \tag{5}$$

where $d_k$ is the propagation delay from the sender of the stream to receiver $k$. Note that existing approaches – such as the specification of a static maxmin allocation for multicast sessions in [21] – also employ the maximum of receiver rates. The distinctive features of the effair allocation are its dynamic nature and accounting for inherent delays in the distributed network.

Our definition for effairness refers to the *effair delivery time* $\phi_k$, which is the amount of time to deliver the data amount $a_k$ from the sender to receiver $k$ under the effair allocation. Now, we formally specify the effair delivery time $\phi_k$ for receiver $k$ by the following expression:

$$a_k = \int_{s_k + d_k}^{s_k + \phi_k} r_k(t)\, dt \tag{6}$$

where $s_k$ is the start time for receiver $k$, $d_k$ is the propagation delay from the sender to receiver $k$, and $r_k(t)$ is the effair rate for receiver $k$. In this definition, $s_k + d_k$ denotes the earliest time when receiver $k$ can obtain data, and $s_k + \phi_k$ represents

the instant when receiver $k$ obtains all its data under the effair allocation.

Before data can reach the receiver and after the receiver has obtained all its data, the effair rate for the receiver is equal to zero:

$$r_k(t) = 0 \quad \text{if} \quad t < s_k + d_k \ \text{ or } \ t \geq s_k + \phi_k. \tag{7}$$

When time $t$ is such that $s_k + d_k \leq t < s_k + \phi_k$, we refer to receiver $k$ as an *active receiver*. The next section presents how to compute the effair rates for active receivers as well as the effair delivery times.

### 3.3 Computation of effairness

As we mention above, we would like to present the effair allocation as a series of static ideal allocations. In this approach, accounting for the impact of delays constitutes the major challenge. Due to the delay differences among streams as well as among the paths to different receivers within the same stream, a change in the rate for one receiver can impact the rate for another receiver *not simultaneously* but with some shift in time.

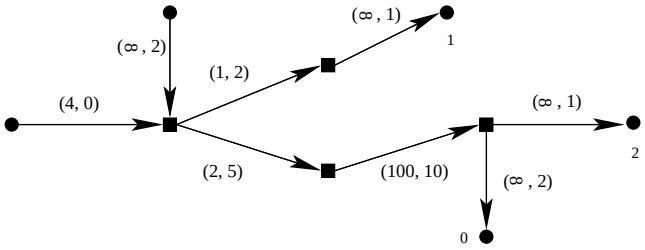
We refer to this shift as an *impact shift* between the *impacting receiver* and the *impacted receiver*. This impact shift can be positive (when the change affects the rate of the impacted receiver in the future) or negative (when the change compels the impacted receiver to modify its rate at a prior moment, as in the case of a receiver that is located closer to the shared edge). Because of impact shifts, the rate allocation at time $t$ can depend on receivers that are not active at this time $t$.

The key idea of our solution is to eliminate this dependence by creating *universal time* such that a change in the rate allocation for one receiver at universal time $\tau$ can affect another receiver only instantaneously, i.e., at the same universal time $\tau$. Such a universal timescale allows us to reduce the problem of finding the effair allocation to the problem of finding a static ideal allocation for the receivers that are active at universal time $\tau$.

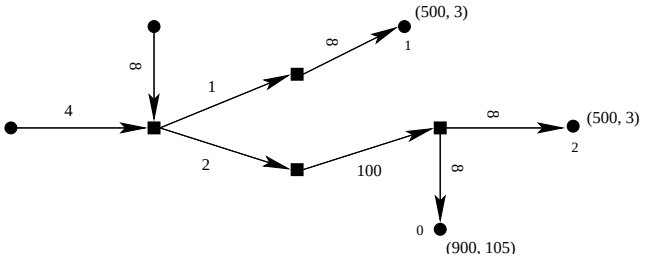
Before we present how to create the universal timescale, note that one can employ our solution only in network configurations where every pair of receivers has at most a single impact shift (i.e., a change in the rate of one receiver does not affect the rate of another receiver at multiple instants). For example, this restriction excludes configurations where sessions traverse the same two links in the reverse order, or where sessions split and then merge again after different propagation delays. Despite these exclusions, most – if not all – network configurations that are commonly used for evaluation of congestion control designs satisfy this condition. The list of the conforming configurations contains dumbbell topologies [1, 3, 9, 16, 25], multiple-bottleneck topologies such as Generic Fairness Configuration GFC-2 [2, 7, 12, 22, 23], and tree topologies used for evaluation of multicast congestion control designs [5, 6, 11, 20, 21].

Let us now address the issues of computing the impact shifts and universal timescale. To represent interactions between streams, we consider the superposition of all the streams in the network.

**Example 2** *Figure 2a shows the superposition of streams for a network with two sessions that are depicted as streams in Figure 1. Since the multicast session traverses both network links of the unicast session, the two corresponding edges – and consequently the nodes adjacent to these edges – of the streams coincide.* ■




(a) superposition of streams




(b) effairness graph

**Figure 2**: Construction of the effairness graph.

In general, the superposition of streams can consist of multiple connected components. Since streams from different components do not share edges (i.e., the corresponding sessions do not share network links), the rate allocation for a receiver in one connected component cannot impact the rate of any receiver in a different component. On the other hand, since a connected component contains a path between every pair of its receivers (note though that some edges on this path can be not from the streams of these two receivers), a change in the rate of one receiver can impact the rate allocations for all the other receivers in the same component.

We compute the *impact shift* between two receivers in a connected component by traversing the path from the impacting receiver to the impacted receiver: set the impact shift to 0 at the impacting receiver; after traversing an edge against its direction, decrease the impact shift by the delay of this edge; after traversing an edge along its direction, increase the impact shift by the delay of the traversed edge. This description

indicates that the impact shift between receivers $k$ and $l$ has the opposite sign and the same absolute value as the impact shift between receivers $l$ and $k$ due to the reversed direction of the traversal.

**Example 3** *Let us use the described procedure to compute the impact shifts between receivers in the network represented in Figure 2a.*

*To reach the bottom multicast receiver* $2$ *from the unicast receiver* $0$ *(see Figure 1 for the topologies of the unicast and multicast streams), one has to go against the edge with the annotation* $(\infty, 2)$ *and then along the edge with the annotation* $(\infty, 1)$*. Thus, the impact shift between the unicast receiver* $0$ *and bottom multicast receiver* $2$ *is equal to* $0-2+1=-1$*. By the reversed traversal of these two edges, we compute the impact shift between receivers* $2$ *and* $0$ *as* $0-1+2=1$*. The values of these impact shifts mean that the rate allocation for the unicast receiver* $0$ *at time* $t$ *can impact the rate for the bottom multicast receiver* $2$ *at time* $t-1$*.*

*The path from the unicast receiver* $0$ *to the top multicast receiver* $1$ *goes against the edges with the annotations* $(\infty, 2)$*,* $(100, 10)$*,* $(2, 5)$ *and then along the edges with the annotations* $(1, 2)$ *and* $(\infty, 1)$*. Hence, the impact shift between the unicast receiver* $0$ *and the top multicast receiver* $1$ *is* $0-2-10-5+2+1=-14$ *while the impact shift between receivers* $1$ *and* $0$ *equals* $14$*.*

*To move from the bottom multicast receiver* $2$ *to the top multicast receiver* $1$*, we need to proceed against the edges with the annotations* $(\infty, 1)$*,* $(100, 10)$*,* $(2, 5)$ *and then along the edges with the annotations* $(1, 2)$ *and* $(\infty, 1)$*. Therefore, the impact shift between receivers* $2$ *and* $1$ *is* $0-1-10-5+2+1=-13$ *while the impact shift between receivers* $1$ *and* $2$ *equals* $13$*. These two impact shifts denote that the rate allocation for the top multicast receiver* $1$ *at time* $t$ *can impact the rate for the bottom multicast receiver* $2$ *at time* $t+13$*.* ■

Now, we define the *universal timescale* for all the receivers. In each connected component, we randomly pick one receiver and refer to it as an *anchor* of this component. The universal timescale for the anchors is the same as their original timescale. For every receiver $k$ that is not an anchor, we move the original timescale of receiver $k$ by the impact shift $i_k$ between this receiver and the anchor of its component; more precisely, we add $i_k$ to every event on the original timescale of receiver $k$. In particular, the instant when receiver $k$ would start receiving its data under the effair allocation appears on the the universal timescale as an *activation time* $\tilde{s}_k$ of receiver $k$:

$$\tilde{s}_k = s_k + d_k + i_k \tag{8}$$

where $s_k$ is the start time for receiver $k$ on the original timescale, and $d_k$ is the delay between receiver $k$ and the sender of its stream.

To represent interactions of streams on the universal timescale, we transform the stream superposition into an *effairness graph* by annotating each receiver $k$ with an ordered two-tuple $(a_k, \tilde{s}_k)$ where $a_k$ is the actual amount of data delivered to receiver $k$. Because these annotations reflect all the delay information needed to compute the effair allocation, the effairness graph omits the edge delays shown by the superposition of streams.

**Example 4** *The superposition of streams in Figure 2a consists of a single connected component. Let us pick the top multicast receiver* $1$ *as the anchor of this component. As Example 3 demonstrates, the impact shift between the unicast receiver* $0$ *and this anchor equals* $i_0 = -14$ *ms while the impact shift between the bottom multicast receiver* $2$ *and the anchor is* $i_2 = -13$ *ms.*

*We create the universal timescale for all the receivers by moving the original timescales for receivers* $0$ *and* $2$ *backward by 14 ms and 13 ms respectively.*

*According to Example 1, the start time for the the unicast receiver* $0$ *is* $s_0 = 100$ *ms while the multicast receivers share their start times:* $s_1 = s_2 = 0$*. Because the delay from the unicast sender to the unicast receiver* $0$ *is* $d_0 = 19$ *ms, the activation time of receiver* $0$ *is equal to* $\tilde{s}_0 = 100+19-14 = 105$ *ms. Since the delays from the multicast sender to the top and bottom multicast receivers are* $d_1 = 3$ *ms and* $d_2 = 16$ *ms respectively, the activation time of the top multicast receiver* $1$ *equals* $\tilde{s}_1 = 0+3+0 = 3$ *ms, and the activation time of the bottom multicast receiver* $2$ *is* $\tilde{s}_2 = 0+16-13 = 3$ *ms.*

*Figure 2b presents the effairness graph received from the stream superposition by omitting the edge delays and annotating the unicast and multicast receivers with (900, 105) and (500, 3) respectively where* $a_0 = 900$ *Kb is the amount of data delivered to the unicast receiver, and* $a_1 = a_2 = 500$ *Kb is the amount of data delivered to each multicast receiver (as specified in Example 1).* ■

On the universal timescale, a change in the rate of one receiver can the affect the rates of the other receivers only instantaneously. This property allows us to express the problem of determining the effair allocation for the original timescale as the problem of computing a static ideal allocation in the effairness graph for each period (on the universal timescale) when the set of active receivers does not change. Well-known solutions for the latter problem [4, 13] provide each receiver with a rate which is a piecewise constant function with the points of possible discontinuity only at the universal time instants when some receiver becomes active or obtains all of its data. Adjusting the difference between the universal and original timescales, we move this piecewise constant function of each receiver $k$ by the impact shift $i_k$ to obtain the effair rate of receiver $k$ on the original timescale.

Figure 3 presents a complete description of our algorithm for computing the effair rates and effair delivery times.

1. In the superposition of all the streams, randomly pick one anchor for each connected component.
2. For every receiver $k$, compute:
   - impact shift $i_k$ between receiver $k$ and the anchor of its component,
   - activation time $\tilde{s}_k$ on the universal timescale,
   - effair rate $\tilde{r}_k(\tau)$ before the activation:
   $$\tilde{r}_k(\tau) = 0 \text{ for } \tau < \tilde{s}_k.$$
3. Construct the effairness graph.
4. Sort the activation times $\tilde{s}_k$ in the increasing order.
5. Set $\delta$ to the smallest activation time.
6. Repeat until determining the effair delivery time $\phi_k$ for each receiver $k$:
   - Compute the ideal static allocation of rates $\rho_l$ to receivers $l$ that are active in the effairness graph at time $\delta$.
   - Set $\theta$ to the earliest of the next activation time and the finish times that active receivers $l$ would have under this allocation.
   - For each receiver $l$ that is active at time $\delta$:
   $$\tilde{r}_l(\tau) = \rho_l \text{ for } \delta \le \tau < \theta.$$
   - For each receiver $m$ that receives all its data at time $\theta$ under this allocation:
   $$\phi_m = d_m + \theta - \tilde{s}_m \text{ and } \tilde{r}_m(\tau) = 0 \text{ for } \tau \ge \theta$$
   where $d_m$ is the delay between receiver $m$ and the sender of its stream.
   - Set $\delta$ to $\theta$.
7. For each receiver $k$, compute the effair rates $r_k(t)$ on the original timescale:
   $$r_k(t) = \tilde{r}_k(t + i_k).$$

**Figure 3**: Algorithm for computation of the effair allocation.

**Example 5** *Let us apply the algorithm from Figure 3 to determine the maxmin effair allocation for the network considered throughout this paper. Actually, the execution for the first three steps of the algorithm is already illustrated in the examples above, and Figure 2b shows the constructed effairness graph.*

*Since the sorted list of the activation times contains* 3, 3, *and* 105, *we initially set* $\delta$ *to 3. At* $\delta = 3$, *only the multicast receivers are active. The maxmin-fair rates for the top multicast receiver* 1 *and bottom multicast receiver* 2 *are respectively* $\rho_1 = 1$ *and* $\rho_2 = 2$. *With these rates, receivers* 1 *and* 2 *would receive their* 500 *Kb of data by time* $3+500/1 = 503$ *and time* $3+500/2 = 253$ *respectively. Because 253 (the smallest of these finish times) exceeds the next activation time 105, we set* $\theta$ *to 105. By time* $\theta = 105$, *receiver* 1 *obtains* $1 * (105 - 3) = 102$ *Kb while receiver* 2 *gets* $2 * (105 - 3) = 204$ *Kb.*

*Now, when we set* $\delta$ *to 105, all three receivers are active. Their maxmin-fair rates* $\rho_0$, $\rho_1$, *and* $\rho_2$ *are equal to* 1. *Under this rate allocation, receiver* 1 *would finish obtaining its remaining* $500 - 102 = 398$ *Kb by time* 503, *receiver* 2 *would finish getting its remaining* $500 - 204 = 296$ *Kb by time* 401, *and the unicast receiver* 0 *would receive its* 900 *Kb by time* 1005. *Since all the receivers are already active, we set* $\theta$ *to 401 which is the smallest of these three finish times. By time* $\theta = 401$, *receiver* 0 *receives* 296 *Kb, receiver* 1 *obtains* 398 *Kb, and receiver* 2 *gets all of its* 500 *Kb. Thus, the effair delivery time* $\phi_2$ *for the bottom multicast receiver* 2 *equals* $\phi_2 = 16 + 401 - 3 = 414$.

*As we set* $\delta$ *to 401, only receivers* 0 *and* 1 *remain active, and their maxmin-fair rates become* $\rho_0 = 2$ *and* $\rho_1 = 1$ *respectively. With these rates, the finish time for receiver* 0 *would be* 703 *while receiver* 1 *would finish receiving its remaining data by time* 503. *Hence, we set* $\theta$ *to 503 which is the smallest of these two finish times. By time* $\theta = 503$, *receiver* 0 *receives* 500 *Kb, and receiver* 1 *obtains all of its data. Therefore, the effair delivery time* $\phi_1$ *for the top multicast receiver* 1 *is* $\phi_1 = 3 + 503 - 3 = 503$.

*After we set* $\delta$ *to 503, only the unicast receiver* 0 *is active. With its maxmin-fair rate of* $\rho_0 = 2$, *receiver* 0 *receives all of its data at time* 703. *Hence, the effair delivery time* $\phi_0$ *for the unicast receiver* 0 *is equal to* $\phi_0 = 19 + 703 - 105 = 617$.

*We summarize the effair rates* $\tilde{r}_0(\tau)$, $\tilde{r}_1(\tau)$, *and* $\tilde{r}_2(\tau)$ *of the receivers on the universal timescale as follows:*

$$\tilde{r}_0(\tau), \tilde{r}_1(\tau), \tilde{r}_2(\tau) = \begin{cases} 0,0,0 & \text{for } \tau < 3 \text{ or } \tau \ge 703, \\ 0,1,2 & \text{for } 3 \le \tau < 105, \\ 1,1,1 & \text{for } 105 \le \tau < 401, \\ 2,1,0 & \text{for } 401 \le \tau < 503, \\ 2,0,0 & \text{for } 503 \le \tau < 703. \end{cases}$$

*Finally, by taking into account the impact shifts* $i_0 = -14$, $i_1 = 0$, *and* $i_2 = -13$ *(determined in Example 3), we calculate the maxmin effair rates* $r_0(t)$, $r_1(t)$, *and* $r_2(t)$ *of the receivers on the original timescale:*

$$r_0(t) = \begin{cases} 0 & \text{for } t < 119 \text{ or } t \ge 717, \\ 1 & \text{for } 119 \le t < 415, \\ 2 & \text{for } 415 \le t < 717, \end{cases}$$

$$r_1(t) = \begin{cases} 0 & \text{for } t < 3 \text{ or } t \ge 503, \\ 1 & \text{for } 3 \le t < 503, \end{cases}$$

$$r_2(t) = \begin{cases} 0 & \text{for } t < 16 \text{ or } t \ge 414, \\ 2 & \text{for } 16 \le t < 118, \\ 1 & \text{for } 118 \le t < 414. \end{cases}$$