A New TCP-Friendly Stabilised CSFQ Mechanism for Layered Multicast

Nick Josevski and Naveen Chilamkurti
Department of Computer Science and Computer Engineering
La Trobe University, Kingsbury, Melbourne, Victoria, Australia-3086

Abstract - Layered Multicasting is an effective technique in distributing large-scale streaming multimedia content over the standard IP networks which form the Internet. Multicasting is effective method whether it is video, voice, or other forms of data when large quantities of receiving users are involved. Multicasting allows a single form of a signal to be transmitted with network protocols managing the mass delivery. The major issue is fairness, which is the desire for the multicast traffic not to overwhelm other usual traffic such as standard TCP traffic. The need for speed of UDP and its designed nature of obtaining as much bandwidth as possible is what causes the issue of negative impact on competing TCP flows. Therefore a goal of operation is to control UDP traffic and use to its potential without disrupting other types of traffic. Shortcomings and any advantageous features of current protocols will be analysed to find mechanisms to assist in improving traffic friendliness. The focus of this dissertation is to analyse aspects of an amalgamation of Core Stateless Fair Queuing and Stabilised Random Early Drop along with other ideas in an attempt to offer an improved scheme for layered multicast fairness that will form a new protocol called Stabilised CSFQ.

1. INTRODUCTION

The heterogeneous nature and scale of the Internet make it impractical for one to one server-client connections when it comes to the need for mass data delivery. When there are a large number of receivers for a multimedia transmission, a collection of individual one to one links coordinated directly by one transmitting system is not practical and cannot be efficiently managed. This along with the growing need for fast real-time distribution of data naturally leads to the idea of using broadcasting for the data delivery. Broadcasting is achieved through routing strategies where a single data stream/signal from a source is 'broadcast' out onto the network and received by active receivers. Broadcasting is the simplest form for delivery but is a clumsy and wasteful by nature with Multicasting being the direction for effective Internet delivery [1].

Such applications growing in popularity that can benefit from a multicast delivery strategy are; voice over IP, teleconferencing, video streaming and web cast for distance learning, Internet games and large scale software distribution. Along with live music and other large events, news and other television like content.

2. MOTIVATION AND RELATED WORK

2.1 CSFQ

Core Stateless Fair Queue (CSFQ) is a distributed approach to fair queuing without introducing processing overhead on core network links [2]. The core component of CSFQ is that only certain routers need to undertake in the special queuing mechanisms while the majority of other routers remain as they are. Therefore only edge nodes (see Fig. 1) undertake congestion control mechanisms allowing core network links to function as normal and without a detrimental effect on existing routing speeds. The protocol is robust and practical due to new processing complexities being added only on the boundaries of the network core. The main distribution backbone of the network (corporate LAN or the Internet) continues to operate at its usual high speed and with a simple queuing scheme. This distributed processing effectively approximates fair queuing without the drawbacks associated in having the FQ mechanism on all routers.

CSFQ approximates fair bandwidth allocation at its edge nodes by utilising the ‘flow id’ field in packet headers. This information is stored in the packets and passed along as they go on their journey. In standard setup core nodes do not use this information, but once the packet reaches the other end of the network where it passes through another set of edge nodes this information can be used again to drop packets or be updated, so capable receivers can also make use of the information. Fig. 1 shows the comparison of standard network layout. On the left is a network where all routers would be operating under fair queuing and on the right a core stateless setup where the three central backbone routers will not need to operate fair queues and can instead undertake in fast first in first out routing.

![Fig. 1: CSFQ Network Structure](image)

For each packet passing along the CSFQ link a forwarding probability \( P \) is calculated. \( P \) is calculated based on the packet label which has the current value of the estimated flow
rate. This label is replaced with a new value each time it passes through a CSFQ link. \( P \) is calculated using equation 1.

\[
P = \min(1, \frac{f}{r})
\]

Where \( r \) is the estimated flow rate and where \( f \) is the configurable fair rate for the link. Therefore the expected rate \( f_{\text{req}} \) of that flow is calculated using equation 2. To determine if the packet should be dropped, the estimated flow rate is compared to the estimated fair rate for that link. If the flow rate exceeds the value of what is determined to be the fair rate, the packet is dropped.

\[
r_{\text{eq}} = r \times \min(1, \frac{f}{r_{\text{fair}}})
\]

\[
\min(f, r_{\text{req}})
\]

### 2.2 Stabilised Random Early Detection (SRED)

Stabilised Random Early Detection (SRED) proposed by Ott et al [3] and expands on RED [4]. It acts to estimate a stateful fair queuing mechanism but without complete flow tracking. This is achieved via a limited degree of tracking which estimates the number of active flows by keeping track of a limited amount of packet history. Recent packets are stored in a fixed size K buffer referred to as a ‘zombie list’; it is composed of packets from K recently seen flows. Extra information is also stored with each element of the zombie list, along with the flow the packet belongs to, a counter and a time stamp. Count starts at zero for the first packet of a flow and is incremented by one each time a new packet arrives that matches the flow. The timestamp field is set to the arrival time of the packet that causes the count increment. RED frequently computes average queue length as part of its packet filtering, whereas SRED does not rely only on current buffer occupancy. SRED follows a random nature like Stochastic FQ [5] to help improve fairness; like SFQ there are immediate benefits obtained from incorporating a random natured filtering mechanism to a structured system.

![Zombie Packet](image)

Fig. 2. Zombie Packet

Flow markers (zombie packets) are the objects that form the zombie list comprised of the flow id, counter, and a timestamp as shown in Fig. 2. Once the zombie list is reached capacity the algorithm begins to filter packets. Now when a packet arrives it is compared to only 1 randomly selected flow marker in the zombie list, and two possible scenarios take place. 1: Hit – arriving packet matches an element in the zombie list, the counter is incremented and the timestamp is changed to the new arrival time. Or 2: No Hit – the two compared do not belong to the same flow, so the existing flow marker based on a probability \( p \) is replaced with the newly arrived packet. The probability of replacement in scenario 2, \( p \) is equal to 0.25, which has been selected through basic tests conducted by the authors. The timestamp is also used to assist in deciding on discarding packets based on their age. The default parameter for the size of the zombie list is 1000.

### 2.3 Current Limitations

There are many existing and proposed mechanisms to offer improved fairness. Majority of them lack enough robust features to have them deployed individually. The lacking features are primarily the ability to have the protocol deployed widely without becoming expensive both in; cost of update and cost of operation to current bandwidth capacities and routing delays. CSFQ offers a solid foundation to address issues of deployment, and has potential to have expanding features added to address its limitations. Such limitations being, its ability to estimate fairness in situations where large traffic flows are present and where such traffic is of short and bursty.

The proposed and existing mechanisms have not been operated in conjunction with layering schemes; they are only discussed in terms of general UDP flows. Layering offers the ability for UDP flows to be more responsive and controllable to alleviate congestion quickly and therefore be fairer to non UDP traffic. This therefore leads to our proposed improvements in section 3. Building on a distributed base, congestion aware and receiver assisted layering techniques can be incorporated with general UDP control schemes to produce a fair and effective protocol.

### 3. Stabilised Core Stateless Fair Queuing (Stabilised CSFQ)

There is a need for a layered multicasting protocol that is congestion aware, friendly to all other traffic, offers distributed processing and is not a major burden on existing networks and traffic patterns. In this paper, we propose a new method of filtering traffic in a distributed fashion that is ‘stateless’ allowing core network routers to operate normally through the use of the CSFQ protocol developed by Stoica et al [2].

Stability is added to CSFQ by means of the Stabilised RED (SRED) protocol [3], where extra mechanisms are added for packet filtering but maintaining a stateless nature of filtering. Further expansions to the original CSFQ structure include specifications for receiver and cooperation in congestion management by means of layered multicast subscription, including stabilizing mechanisms in layer changes. The focus of this protocol is to be distributed, stable and fair to all participating agents. The aim is to improve the operation of CSFQ to make it more TCP-friendly via stable means, therefore leading to the title Stabilised CSFQ.

SRED offers a useful means to expand the core operation of CSFQ in assisting in filtering at edge nodes. Where SRED is not suitable to operate on all network links it can be deployed
using the CSFQ structure on only edge networks. The ‘zombie list’ filtering technique in conjunction with base CSFQ fairness checks is planned to offer an improved fair filtering at the edge of networks. Receiver side additions will assist in determining the state of congested links and better interpret lost packets. Therefore in a layered multicast system the enhancements discussed allow receivers to participate with ease to achieve fair link utilization. With the added features in edge nodes and also receiver side to assist in congestion management, Stabilised CSFQ will be analyzed via simulations to show the predicted improvements to fairness.

In this section general goals for Stabilised CSFQ will be discussed followed by more technical details of operation and the amalgamation of components into basic CSFQ. Section 4 will discuss simulation topologies and analyze results obtained.

3.1 Objectives

One of the objectives of Stabilised CSFQ is to obtain a fair bandwidth usage for layered multicast traffic over networks where many competing non-UDP flows are present and are not to be deprived of their entitled fair share of bandwidth. This includes network fairness mechanisms that are not excessively restrictive; as to permit a link to become under utilized from excessive packet dropping. Stabilised CSFQ further emulates complete fair queuing without the associated overheads. With receiver participation through layer joining and leaving the fairness objectives can be met.

4. Simulation Setup

Stabilized CSFQ is implemented using discrete event simulator known as Network Simulator (NS-2) [6]. In order to simulate Stabilized CSFQ, We have added our own module on the top of C++ and OTcl CSFQ source code created by Stoica et al [2], [7] along with enhancements from SRED [3]. The enhancements are discussed in Section 3 along with the original concepts that are inherited from CSFQ and SRED. We ran a number of simulations to test simple queuing mechanisms with few flows to number of flows with CSFQ and SRED algorithms. Finally we tested layering and varying subscription changes based on the UDP receivers with Stabilized CSFQ. Various combinations of flow counts, topologies, start and finish times have been used to simulate and analyze a wider range of results. These experiments also give us a picture of how the algorithms respond to large amount of UDP flows.

4.1 Topologies

We used three topologies to compare, simulate and analyze Stabilized CSFQ.

Topology 1:

This topology is the simplest, offering a basic setup for the introduction of queuing mechanisms and to be used to create base lines for comparisons. The servers images (left) are the data sources, there is only one receiver attached by one link to the only router. The single router with different queuing mechanisms will be the bottleneck and will be point where we will monitor to determine throughput of flows.

Fig. 3. N to 1 Topology

Topology 2:

This topology is used for the majority of experiments we conducted. It offers a collection of N transmitters at source side and N number of receivers at receiver end. There are four routers connecting sources and receivers. Any one of the three links between four routers can be selected as a bottleneck and these routers can have CSFQ, SRED or Stabilised CSFQ queues. For the division of edge and core the central most links becomes the core of the network where standard FIFO [8] queue mechanisms can be used.

Fig. 4. Dumbell topology

Topology 3:

Topology 3 is used to understand performance of Stabilised CSFQ with multiple flows and different varieties of sources. As shown in Fig. 5, central backbones with intervening flows create congestion on all links. The UDP transmission must travel across all the three backbone links. TCP flows can each cross one backbone link interfering with the UDP transmission, so the bandwidth of the TCP flows across those links can be observed. Measuring and averaging the TCP share on each of the three congested links will offer data for fairness measures. A clear path of TCP flow is shown in Fig. 5.
4.2 Varying Flow Quantities and Ratios

In these experiments, we used topology 2 to compare CSFQ with Stabilised CSFQ with varying UDP and TCP flows. The goal of these experiments is to give a broader view of varying TCP share as flow counts of both UDP and TCP are varied. The simulation runs for 120 seconds and all flows begin transmitting at time zero ($t_{\text{start}} = 0$ seconds) and continue until two minutes ($t_{\text{end}} = 120$ seconds). The TCP window is set to 100 to allow TCP to attempt and gain as much fair bandwidth as it can. The UDP and TCP flow counts are varied to determine the fairness of stabilized CSFQ with respect TCP-share.

A summary of the data is presented in Table 1 along with a further summary in the form of percentage comparisons in Table 2. In Table 1 ideal share is shown as a baseline in a perfect split based on flow amounts.

Table 1. Varying Flow Count Simulation Summary

<table>
<thead>
<tr>
<th>Flow Counts</th>
<th>UDP</th>
<th>TCP</th>
<th>CSFQ Received Data (kb)</th>
<th>IDEAL SHARE</th>
<th>Stabilised CSFQ Received Data (kb)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Total UDP</td>
<td>Total TCP</td>
<td>IDEAL UDP</td>
<td>IDEAL TCP</td>
<td>Stabilised UDP</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>10</td>
<td>18427</td>
<td>5355</td>
<td>50.00%</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>30</td>
<td>16804</td>
<td>6916</td>
<td>50.00%</td>
</tr>
<tr>
<td>5</td>
<td>10</td>
<td>30</td>
<td>14281</td>
<td>9344</td>
<td>33.33%</td>
</tr>
<tr>
<td>10</td>
<td>5</td>
<td>15</td>
<td>19003</td>
<td>3034</td>
<td>66.67%</td>
</tr>
<tr>
<td>8</td>
<td>10</td>
<td>15</td>
<td>8146</td>
<td>15247</td>
<td>21.05%</td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td>12</td>
<td>16404</td>
<td>7301</td>
<td>50.00%</td>
</tr>
<tr>
<td>14</td>
<td>14</td>
<td>15</td>
<td>15958</td>
<td>7726</td>
<td>50.00%</td>
</tr>
<tr>
<td>15</td>
<td>15</td>
<td>10</td>
<td>14428</td>
<td>9199</td>
<td>50.00%</td>
</tr>
<tr>
<td>30</td>
<td>30</td>
<td>30</td>
<td>14408</td>
<td>9221</td>
<td>50.00%</td>
</tr>
</tbody>
</table>

As noticed in Table 4, increasing number of UDP flows were having a negative effect on the TCP-share. The increased UDP flow count was detrimental regardless of the TCP count. When UDP flows reached 15 the TCP share drops as compared to the CSFQ. On the contrary the Stabilised CSFQ shows a good sharing ratio (around 6.5%) at 12 TCP flows.

As in this scenario TCP and UDP flows are traveling over a 2Mb bottleneck link and all the UDP flows are attempting to transmit each at a rate close to 2Mb. At this stage the level of congestion control of TCP would be detecting lost packets and will be acting to alleviate the bottleneck by reducing rates. This leads us to our final set of simulations, where receivers will be reducing rates to offer a more robust test case of the protocols potential for fairness.

4.3 Receiver Convergence under Stabilised CSFQ

The previous experiments show that Stabilised CSFQ operates well in improving fairness for TCP traffic competing with a number of UDP flows. The previous simulations do not involve any receiver side operations.

In order to test receiver cooperation in heavy congestion situations, in this simulation UDP receivers will unsubscribe layer subscription levels (UDP flows) at various times to emulate receivers responding to congestion.

![Fig. 6. Receivers slowly joining layers – Stabilised CSFQ](image)

The setup for the simulation in this scenario involves a handful of receivers behind the 2Mb bottleneck links, and 30 TCP flows operating from 0 to 12 seconds. Twelve seconds was chosen to show the convergence time in the first few seconds. Fig. 6 shows the total summary values of traffic passing through the bottleneck link. TCP traffic is high at the beginning as the UDP receivers are joining layers slowly. Approximately 2.4 seconds into the simulation, a fair level of convergence is reached where the layer count for the UDP receivers is around 8 or 9 layers of the 15 layers available. In order to compare our results with standard CSFQ, we ran additional simulations and the results are shown in Fig. 7. In this simulation standard CSFQ is subjected to the same conditions as with stabilized CSFQ, to offer a comparison of the two in a more theoretical layered multicast scenario.
The flow share difference between Fig. 6 and Fig. 7 is clearly visible and shows that Stabilised CSFQ is operating to achieve more friendliness to other flows over standard CSFQ.

5. CONCLUSIONS AND FUTURE WORK

In this paper we analyzed existing mechanisms currently proposed for improving fairness controls applied to UDP flows that are non responsive to congestion by nature. We discussed the limitations and advantages of current protocols and proposed enhancements for current approaches amalgamating ideas under the name Stabilised CSFQ.

The first scenarios involve varying flow quantities of TCP and UDP flows and measure their fair share with and without Stabilized CSFQ. The second scenario is as setup to observe the layer subscription of receivers at various time intervals. These test scenarios offer proof of its ability to achieve fairness, particularly those simulations which compare its ability against other mechanisms. The proof is offered as values of performance increases where Stabilised CSFQ outperforms the other strategies in these simulations.

In our future work, firstly, we think that more comprehensive tests would allow for detailed analysis of packet dropping and analysis of which flows were affected. Next we think modifications could be performed to optimize the zombie list, by changing the size of the zombie list, the conditions for packet dropping, including incorporating time based factors for dropping old zombie packets. Packets would be discarded from the list more frequently and the list made to operate more dynamically based on changing network situations.

Finally, there is potential for improving the ability of the receiver to better judge layers through modifying and refining the TCP throughput equation to suit how the network manages congestion. This can be achieved through more simulation tests involving non uniform traffic conditions and greater variations in active flows.

6. REFERENCES


