Impacts of the Startup Behavior of Multilayered Multicast Congestion Control Protocols on the Performance of Content Delivery Protocols

Christoph NEUMANN   Vincent ROCA
INRIA Rhône-Alpes, Planète research team, France
christoph.neumann|vincent.roca@inrialpes.fr
http://www.inrialpes.fr/planete/

Abstract

The FLUTE file delivery application (RFC 3926) and the underlying massively scalable reliable multicast protocol, ALC (RFC 3450), are used for the scalable distribution of contents in a broadcasting system having no feedback channel. When used in a routed network (like the Internet), a layered congestion control protocol is required. This paper analyzes the impacts of the startup behavior of several such protocols on the performance of multicast content delivery protocols through mathematical models. Our results show (1) that these congestion control protocols have a major impact on the download time, and (2) that the three protocols considered in this study yield significant differences.

Keywords: Reliable Multicast Content Delivery, FLUTE, ALC, Layered Congestion Control Protocol

1 Introduction and Related Works

1.1 Motivations

Using a reliable multicast content delivery system for the distribution of popular content to a large set of clients has shown to be very effective. The FLUTE file delivery application (RFC 3926[1]) and the underlying massively scalable reliable multicast protocol ALC (RFC 3450 [2]) are increasingly used to this purpose, and they are included in the technical specifications of the 3GPP MBMS service [3] and of the DVB-H IP Datacast service [4]. Thanks to them, a content can easily be delivered to several millions of clients, through a carousel: the content is continuously broadcast or multicast, and interested clients can join the session, download the content and leave the session whenever they want.

The same approach should be used in the Internet (or any multicast capable routed network, e.g. within a site) when a content has to be broadcast to a large number of receivers. Even if multicast routing deployment is far behind expectations, the increasing number of potential receivers, in particular (but not limited to) research networks, and the availability of high bandwidth access technologies (e.g. ADSL/ADSL2) makes this technology attractive. Content delivery systems using FLUTE are already deployed (e.g. within the M6BONE network) and we can expect it to continue.

A major question remains: how long does it take for a client to successfully download the content? Several parameters have to be taken into account: the content size, the available network bandwidth, the Forward Error Correction code features, and the packet loss model. When used in a *routed network* (like the Internet), a layered congestion control protocol is mandatory. In that case, another parameter is of importance: the congestion control protocol behavior during the startup phase, from the moment a client joins an ALC session and the moment an equilibrium is reached. Indeed, because of this congestion control protocol, on session startup, the client's reception rate progressively increases until it reaches a "fair share" of the available bandwidth between the source and the client (the exact fairness definition depends on the protocol used and is out of the scope of the present paper). The time required to reach the steady rate is not so small as one would expect and has a major impact on reception performance. This is especially true when a client only downloads a small to medium size content, since the time required to reach the target rate will dominate the global download time.

In this work we investigate the impacts of the startup behavior of three layered multicast congestion control protocols on reception performance. We first introduce a mathematical model and then analyze the amount of data actually received. We assume in particular that receivers do not experience any loss, i.e. have not yet reached their target rate.

1.2 Asynchronous Layered Coding (ALC)

The Asynchronous Layered Coding (ALC) protocol (RFC 3450) [2] of the IETF RMT working group is a layered reliable multicast protocol for scalable content delivery. ALC is well suited to the transmission of popular content in an “on-demand” mode, where clients join an ALC session, retrieve data, and leave at their own discretion. This is made possible by the large use of FEC (Forward Error Correction) encoding [5], and by the transmission of all the packets (source and parity) continuously (and often in a random order) on the various ALC layers [6].

1.3 Layered Congestion Control Protocols

In a routed network, each receiver adapts dynamically how many layers to receive, depending on its access network and on competing traffic. This receiver-driven decision is taken by an associated TCP-friendly layered congestion control protocol (e.g. RLC [7], FLID-SL/DL [8] [9], or WEBRC [10] [11]). The client behavior will also largely differ according to the protocol used. For instance transmissions will take place either at some fixed predefined bit-rate on each layer (RLC, FLID-SL), or using a cyclic, dynamically changing bit-rate (FLID-DL, WEBRC).

The IGMP leave latency (delay between when the last receiver of a LAN leaves a multicast group and its effect) is of importance. This latency is usually 3 seconds but can be higher depending on the IGMP implementation. The IGMP leave latency is known to affect the behavior of a statically layered protocols like RLC or FLID-SL. The only exceptions are the FLID-DL and WEBRC protocols that counteract this latency thanks to a dynamic layering approach.

We now give a short overview of these protocols.

1.3.1 RLC

In Receiver-Driven Layered Congestion Control (RLC) [7] transmissions take place at constant bit-rate on each layer. A receiver experiencing no loss can join higher layers at so called ”increase signals” (indicated in the packet header). This mechanism helps to coordinate the behavior of receivers behind bottleneck links. The distribution of these signals and the transmission rate of each layer are specified in section 2.1. If the receiver experiences losses he drops the highest layer. Then a ”deaf period” (of duration at least equal to the IGMP latency) is observed, during which a receiver ignores losses and increase signals. Its goal is to let the network prune the congested branch and then settle. After that the receiver can again drop layers if there are further losses or on the opposite add layers at the next increase signals if there is no loss any more.

1.3.2 FLID-SL

Fair Layered Increase/Decrease with Static Layering (FLID-SL) [8], like RLC, relies on constant rate transmissions on each layer. The server also places signals into packets that indicate receivers when to join layers. However in FLID-SL these signal are based on a probabilistic function that tells whether to place or not an increase signal into a packet. The details of this function and the transmission rate of each layer are specified in section 2.2. Another difference is that the probing mechanism of RLC (used to test if joining a new layer is likely to be feasible or not) is no longer used in FLID-SL.

1.3.3 FLID-DL

Fair Layered Increase/Decrease with Dynamic Layering (FLID-DL) is a dynamic layered version of FLID-SL where the bit rate of each session layer is changing dynamically in order to avoid the problems of the IGMP leave latency that static layered protocols like RLC or FLID-SL have. The idea is to continuously decrease transmission rate of each layer until reaching a point where no data is transmitted on this layer. After some time the layer restarts transmitting at maximum rate and again continuously decrease transmission rate. The receiver has to continuously join new layers in order to keep his reception rate constant. A receiver does not has to leave a layer in order to decrease reception rate, but just does nothing. Therefore the effect of the IGMP leave latency is bypassed.

However FLID-DL needs a prohibitive amount of overhead traffic, in particular because of the high number of IGMP messages. Therefore we focus on WEBRC, a protocol that also uses a dynamic layered approach (like FLID-DL) without its drawbacks.

1.3.4 WEBRC

Like FLID-DL, Wave and Equation Based Rate Control (WEBRC) uses the idea of dynamically changing transmission rates of each layer. In WEBRC the transmission rate of each layer evolves in waves. It is periodic, with an exponentially decreasing rate during the active period followed by a quiescent period. This feature reduces the number of layers needed (compared to FLID-DL), which also reduces the IGMP overhead, while solving the IGMP leave latency problem.

Unlike the congestion control protocols we have seen so far, that all rely on the losses observed by the receiver, WEBRC is equation-based. The available bandwidth is calculated by each receiver using an equation that imitates TCP behavior, and each receiver then adapts its reception rate accordingly.

The remainder of this paper is organized as follows: In section 2 we give the mathematical model of the startup phase of each congestion control protocol. In section 3 we explain how to use our results in practice. Finally section 4 concludes.

2 Analysis of the Startup Phase

2.1 Startup Behavior with RLC

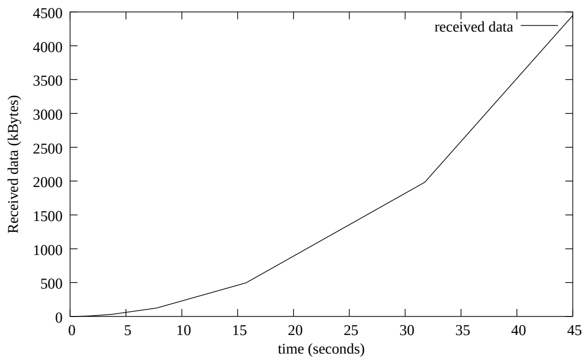

Figure 1. Amount of data received in the startup phase of RLC.

In RLC [7] a receiver experiencing no loss can add a new layer upon the reception of a dedicated "increase signal". These signals are exponentially distributed over the layers, using a factor of two, making the opportunities of adding higher layers less frequent than with lower layers. The minimum delay, t_l, after which layer l can be added, if no loss occurs, is: $t_l = (2^l - 1)t_0$, where t_0 is a fixed period (we use $t_0 = 0.25$ seconds in our experiments). Then each "increase signal" of layer l is repeated with a period: $T_l = 2^l t_0$. With RLC, the transmission rate of each layer is twice the rate of the previous layer. Therefore, when dropping a layer after a packet loss, this scheme performs a TCP-like multiplicative decrease. But adding a layer also doubles the reception rate. Since the increase signals are exponentially distributed over the layers, this is not compliant with the TCP slow-start algorithm which follows an initial exponential behavior (TCP doubles the transmission rate each RTT in the startup phase).

The transmission rate of layer $l \in \{0; alay\_nb - 1\}$, where $alay\_nb$ is the total number of layers in an ALC session, follows a doubling scheme:

$$b_l = \begin{cases} b_0 & \text{if } l = 0 \\ 2^{l-1} b_0 & \text{if } l \geq 1 \end{cases}$$

Therefore, the amount of data received through a single ALC session in the startup phase, at time $t = i * t_0$, multiple of the RLC's time slot period, is:

$$\begin{aligned} & Rx(t = i * t_0) = \\ & data\_received\_on\_base\_layer \\ & + \sum_{l \in \{active\_upper\_layers\}} data\_received\_on\_layer\ l \\ = \ & b_0 \sum_{k=0}^{i-1} t_0 + \sum_{l:\, l>0\ and\ 2^l < i+1} 2^{l-1} b_0 \sum_{k=2^l-1}^{i-1} t_0 \\ = \ & b_0 t_0 \left(i + \sum_{0<l<\log_2(i+1)} 2^{l-1}(i + 1 - 2^l) \right) \end{aligned}$$

If we only consider the moments when a new layer is added, i.e. if $\exists\ L : i + 1 = 2^L$, then this equation can be simplified:

$$Rx(t = i * t_0) = \frac{i(i+2) b_0 t_0}{3} \qquad (1)$$

Figure 1 shows the $Rx(i)$ curve as a function of time, when $b_0 = 11.9$ kbps and $t_0 = 0.25$ sec.

2.2 Startup Behavior with FLID-SL

We now consider the FLID-SL (Static Layer) congestion control protocol [8] which shares many similarities with RLC. The b_l can follow a multiplicative scheme:

$$b_l = \begin{cases} b_0 & \text{if } l = 0 \\ (C^l - C^{l-1}) b_0 & \text{if } l \geq 1 \end{cases}$$

where $C > 1$ is the multiplicative factor. [8] and [9] recommend to use $C = 1.3$.

The main difference between FLID-SL and RLC concerns the period T_l between two "increase signals" on layer l. T_l depends on a probabilistic function p_l which indicates the probability to increase the subscription layer in each time slot. On average T_l is given by:

$$T_l = \frac{TSD}{p_l}$$

where TSD is the Time Slot Duration. [8] proposes a heuristic for p_l to mimic TCP. [9] suggests a simpler scheme that we consider here:

$$p_l = min\left(1.0, \frac{20 * pkt\_sz * TSD}{C^l * b_0}\right)$$

We have to consider the two parts of this equation: for small l where $\frac{20*pkt\_sz*TSD}{C^l b_0} > 1.0$, and for l where $\frac{20*pkt\_sz*TSD}{C^l b_0} \leq 1.0$.

Let l_{min} be such that: $\frac{20*pkt\_sz*TSD}{C^{l_{min}} b_0} = 1.0$.

$$T_l = \begin{cases} TSD & \text{for all } l \leq l_{min} \\ \frac{TSD}{p_l} = C^l \frac{b_0}{20*pkt\_sz} & \text{for all } l > l_{min} \end{cases}$$

We can now divide the formula for the amount of received data into two parts.
For all layers l where $l \leq l_{min}$:

$$Rx_1(t) = t * b_0 + \sum_{l=0}^{l_{min}} ((t - l * TSD) * (C^l - C^{l-1}) * b_0)$$

For all layers $l = x + 1$ where $l > l_{min}$:

$$Rx_2(t) = \sum_{x=l_{min}}^{l_{max}} ((t - l_{min} * TSD - \sum_{y=l_{min}}^{x} (T_y)) * (C^{x+1} - C^x) * b_0)$$

with l_{max} such that: $t - l_{min} * TSD - \sum_{x=l_{min}}^{l_{max}} (T_x) = 0$.

The overall amount of received data is then:

$$Rx(t) = Rx_1(t) + Rx_2(t) \quad (2)$$

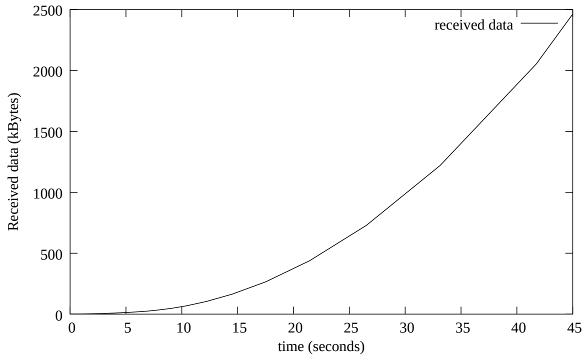

Figure 2. Amount of data received in the startup phase of FLID-SL.

Figure 2 shows the $Rx(t)$ curve as a function of time, when $C = 1.3$, $b_0 = 11.90$ kbps, and $pkt\_sz = 576$ bytes. We use rounded values in our calculations for l_{min} and l_{max}.

We see that FLID-SL and RLC give similar results, even if FLID-SL, with the default parameters suggested in the literature, leads to a slower reception rate progression. We do not consider the FLID-DL (Dynamic Layering) scheme [8] here, since WEBRC replaces FLID-DL favorably.

2.3 Startup Behavior with WEBRC

WEBRC behaves differently than RLC or FLID-SL and is capable of adding layers much faster during the startup phase. Indeed the reception speed is multiplied by a factor of $C = \frac{4}{3}$ each epoch (by default 0.5 second), creating an exponential increase (TCP does the same since the transmission rate is doubled each RTT during the first stage of the slow-start algorithm), and then stabilizes around a "reasonably " fair share of the available bandwidth, determined through a TCP throughput equation.

More precisely, every $epoch$, the reception rate is increased by a factor C. The amount of data received through a single ALC session in the startup phase, at time $t = i * epoch$, multiple of the $epoch$ time slot period, is:

$$Rx(t = i * epoch) = \sum_{l=0}^{i-1} C^l * b_0 * epoch$$

By resolving the sum we obtain:

$$Rx(t = i * epoch) = \frac{(C^i - 1)b_0 * epoch}{C - 1} \quad (3)$$

This equation highlights the exponential increase of the reception rate (in $(\frac{4}{3})^i$ instead of i^2 with RLC).

3 How to Use these Results in Practice? Lessons Learned

3.1 Comparison When Ignoring the Target Rate Limitation

Our models allow us to calculate the total amount of data received at a given time when taking only into account the startup phase, before the target rate (assumed to be close to the TCP equivalent share of the bandwidth) is reached. Figure 3 compares this amount of received data using formulas 1 (RLC), 2 (FLID-SL) and 3 (WEBRC). It shows that WEBRC is the less impacted by the startup phase. A receiver benefits from the exponential behavior of the reception rate (and quickly reaches the TCP equivalent throughput). On the opposite, FLID-SL and RLC both experience a very slow reception rate increase. Therefore, *during small sessions, for instance when a client downloads a small file, RLC and FLID-SL download performance will essentially be dominated by the protocol startup behavior, unlike WEBRC.*

3.2 Comparison When Considering the Target Rate Limitation

Let's now assume that the target rate limits the reception rate to 10 Mbps, which is a reasonable value within a site network, or in new ADSL2 access networks, which are being more and more deployed.

In that case, a receiver's download session behavior will be first controlled by the startup phase, and once the aggregate 10 Mbps threshold is reached (if ever this is the case!), download will then take place at this total rate [1]. It is important to consider this global reception bandwidth limitation

[1]We ignore here any target rate fluctuation or congestion control protocol limitation that may prevent a receiver to keep this rate since the present paper only focuses on the startup behavior.

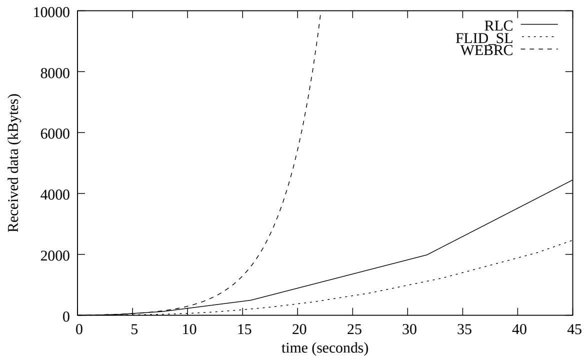

Figure 3. Amount of data received in the startup phase of WEBRC, RLC and FLID_SL. No bandwidth limitation.

since WEBRC in practice quickly reaches this threshold, which is not considered in section 3.1. The times when this target rate is achieved and the corresponding number of layers can easily be calculated using the b_l transmission rate formulas of section 2 first, and then the t_l formulas at which the layer is added.

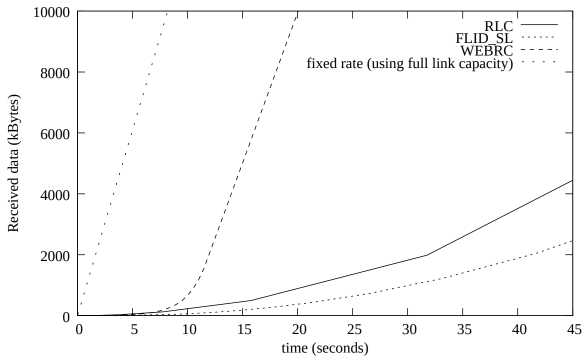

Figure 4. Amount of data received in the startup phase of WEBRC, RLC, and FLID_SL. Fixed 10 Mbps maximum bandwidth limitation.

This is shown in figure 4, where we use the same parameters as in section 2. The amount of received data with WEBRC, RLC and FLID-SL is also compared to a fixed rate 10 Mbps reception scheme, in order to better appreciate the global impact of the respective congestion control startup phases. WEBRC needs only ≈ 11.5 seconds to reach the target rate bandwidth, then the global reception rate remains constant. RLC requires ≈ 63.7 seconds, and FLID-SL ≈ 495.0 seconds (not shown in the figure). The number of layers needed to reach this bandwidth is respectively 9 layers with RLC and 25 layers with FLID-SL.

A Small Example:

Assume a client is downloading a 500 kByte file. The minimum reception time amounts to ≈ 15.75 seconds with RLC, ≈ 22.51 seconds for FLID-SL and ≈ 9.44 seconds for WEBRC. On the opposite, if no congestion control is used (which is never recommended!), receiving at 10 Mbps, $\frac{500kBytes}{10Mbps} \approx 0.41$ seconds are sufficient to download the content! Figure 4 and this small example clearly show the *large performance gap between the different congestion control protocols and a fixed rate transmission*, and confirms the major performance differences between the various protocols.

Comparison with TCP:

[12] introduces formulas to model TCP with its slow start behavior. With an RTT=200ms we obtain a download time of ≈ 2.29 seconds for the same 500kByte content. Therefore TCP is faster than all multilayered multicast congestion control protocols, which is not surprising since sender and receiver are closely synchronized.

3.3 Transmission Rate Granularity Considerations

From the above tests, one must not too quickly conclude that FLID-SL should definitely be banned. The RLC protocol reaches the target rate faster than FLID-SL because it uses a smaller number of layers, each of them having a higher transmission rate. This is efficient, but it also compromises the granularity with which a receiver can adapt to congestion situations. RLC reception rate granularity (factor 2) is far too coarse to enable an appropriate behavior compared to FLID-SL (factor 1.3 only), and adding a single layer with RLC can lead to a severe network congestion. This is why *FLID-SL is preferred over RLC in practice*.

3.4 Reception Inefficiency Factors

In practice ALC introduces several inefficiencies, because of:

- the packet scheduling scheme, since the same packet may be received on several layers. This issue may be eliminated if an extensible FEC code is used [5] since an infinite number of fresh parity packets may be generated. Since in practice public implementations of those FEC codes not available (they are protected by many IPRs), the duplication problem remains and is all the more acute as the number of layers within the session is large;

- FEC inefficiencies, for instance caused by the coupon-collector problem, if a small block FEC codec is used, or by the intrinsic FEC inefficiency, if one of the various LDPC codes is used [13].

These factors must be considered if one desires to estimate the actual download time, since a few percents of packets in addition to the object size will be required. How large this additional time is, depends on the inefficiencies mentioned above. It is impossible to predict it in general since it depends on many configuration-specific parameters.

4 Conclusions

We have compared the startup phase of three layered congestion control protocols, RLC, FLID-SL, and WEBRC, and we have introduced formulas that enable to calculate the amount of data received by a client at any time, before reaching the target reception rate. The present paper has shown that:

- the congestion control startup phase has major performance impacts *with RLC and FLID-SL, where this phenomenon dominates the reception rate during several tens of seconds*. This behavior is the result of the reactive approach used by these protocols (a client adds layers until it experiences packet losses that are interpreted as the sign of network congestion), and the rate at which layers can be added is deliberately limited in order to limit the possible congestion it may create when a client download rate approaches its fair share.

- reaching the target rate faster by using a smaller number of layers, each of them having a higher transmission rate, is efficient. This is why RLC (that uses a 2.0 scaling factor) yields higher performances than FLID-SL (that uses a 1.3 scaling factor). But this solution also largely compromises the granularity with which a receiver can adapt to the equivalent TCP throughput, and *in practice FLID-SL is preferred over RLC*.

- *WEBRC is less impacted*, essentially because of its initial exponential rate increase. This is made possible by the equation-based approach of this protocol and a dedicated "slow start" mechanism (which is not so slow when compared to RLC/FLID-SL). Since a receiver cannot calculate a meaningful target rate from its measurements, it uses default values (which warrants the fast reception rate increase we noticed) and leaves this mode on some events (e.g. a packet loss, a sharp increase in the multicast RTT, or some inconsistency in the experienced reception rate).

Yet this work does not take position on the respective merits of the three congestion control protocols, and in particular on their ability to compete fairly with TCP and similar congestion-controlled sessions. In particular the fact that RLC and FLID-SL do not take any account the source/receiver RTTs is an intrinsic limitation.

Acknowledgements

This work has been made possible by the e-Next network of excellence.

References

[1] T. Paila, M. Luby, R. Lehtonen, V. Roca, and R. Walsh, *FLUTE - File Delivery over Unidirectional Transport*, Oct. 2004, Request for Comments 3926.

[2] M. Luby, J. Gemmell, L. Vicisano, L. Rizzo, and J. Crowcroft, *Asynchronous Layered Coding (ALC) protocol instantiation*, Dec. 2002, Request for Comment 3450.

[3] *3rd Generation Partnership Project; Technical Specifi cation Group Services and System Aspects; Multimedia Broadcast/Multicast Setvice; Protocols and Codecs (Release 6)*, Nov. 2004, 3GPP TS 26.346 v1.5.0.

[4] *IP Datacast Baseline Specifi cation: Specifi cation of Interface I MT (a080)*, Apr. 2004, DVB Interim Specifi cation.

[5] M. Luby, L. Vicisano, J. Gemmell, L. Rizzo, M. Handley, and J. Crowcroft, *The use of Forward Error Correction (FEC) in reliable multicast*, Dec. 2002, Request for Comments 3453.

[6] J. Byers, M. Luby, M. Mitzenmacher, and A. Rege, "A digital fountain approach to reliable distribution of bulk data," in *ACM SIGCOMM'98*, Aug. 1998.

[7] L. Vicisano, L. Rizzo, and J. Crowcroft, "Tcp-like congestion control for layered multicast data transfer," in *IEEE INFOCOM'98*, Feb. 1998.

[8] J. Byers, M. Frumin, G. Horn, M. Luby, M. Mitzenmacher, A. Roetter, and W. Shaver, "Flid-dl: Congestion control for layered multicast," in *2nd Workshop on Networked Group Communication (NGC2000)*, Nov. 2000.

[9] M. Luby, L. Vicisano, and A. Haken, *Reliable multicast transport building block: Layered Congestion Control*, Nov. 2000, Work in Progress: <draft-ietf-rmt-bb-lcc-00.txt>.

[10] M. Luby, V. Goyal, S. Skaria, and G. Horn, "Wave and equation based rate control using multicast round trip time," in *ACM SIGCOMM'02*, Aug. 2002.

[11] M. Luby and V. Goyal, *Wave and Equation Based Rate Control Building Block*, Apr. 2004, Request for Comments 3738.

[12] James E. Kurose and Keith W. Ross, *Computer Networking - A Top-down Approach Featuring the Internet*, Addison-Wesley, Reading, Massachusetts, third edition, 2005.

[13] V. Roca and C. Neumann, "Design, evaluation and comparision of four large block fec codes, ldpc, ldgm, ldgm staircase and ldgm triangle, plus a reed-solomon small block fec codec," Research Report 5225, INRIA, June 2004.