# Guaranteed packet delays with network coding

Ali Mahmino, Jérôme Lacan, Christian Fraboul

# Guaranteed Packet Delays with Network Coding

Ali Mahmino
ISAE and INPT, Univ. of Toulouse,
1, place E. Blouin,
31056 Toulouse Cedex, France.
Email: ali.mahmino@isae.fr

Jérôme Lacan
ISAE/LAAS-CNRS, Univ. of Toulouse,
1, place E. Blouin,
31056 Toulouse Cedex, France.
Email: jerome.lacan@isae.fr

Christian Fraboul
INPT, Univ. of Toulouse,
2, rue Camichel - BP 7122
31071 Toulouse, France
Email: christian.fraboul@enseeiht.fr

***Abstract*— In the context of networks providing QoS guarantees, the end-to-end delay experienced by a packet is an important parameter. In this paper, we show that network coding can be used to decrease worst case end-to-end bounds when compared to a classical routing strategy. This result can be explained by the fact that network coding can cope with congestion better that classical routing due to its property to process simultaneously packets from different flows. In this paper, two network coding strategies, applied to networks providing QoS guarantees, are presented. We present an evaluation of worst case delays both in routing and coding approaches with network calculus tools. An interesting result is that network coding can improve these guaranteed end-to-end bounds even in network topologies where the throughput is not improved.**

## I. Introduction

In networks providing quality of service (QoS) guarantees, input data flows verify constraints of burstiness and maximal throughput. On the other hand, the network ensures a level of quality of service guarantee in terms of end-to-end delays or throughput. The different guarantees and constraints characterizing the network and the flows can be represented by using the network calculus framework [1] which allows the computation of upper bounds in terms of delays, throughput or buffer sizing.

In this paper, we focus on decreasing worst case end-to-end delays by using network coding techniques. Network coding is a concept introduced in [2], which allows routing nodes to perform linear combinations of input packets instead of simply forwarding them. As opposed to random or opportunistic network coding approaches (see respectively [3] and [4]), we consider in this work a classical approach (see *e.g* [5] or [6]) where the nodes combine the packets following a fixed network code determined *a priori*. This approach, which is theoretically the most efficient, could lead to buffering in coding nodes and then, to an end-to-end delays increase. For independent Poisson flows, the average queue sizes and the delays were studied in [7]. Another analysis of the gain in delay performance resulting from network coding was proposed in [8] for the download of whole files.

In our context, the guarantees on end-to-end delays must be ensured for each packet. For this problem, the main interest of network coding is that a coding node processes several input packets simultaneously and thus, it allows to reduce the maximum time spent by packets in the buffers when compared to a classical routing approach dealing with packets in sequence. In counterpart, other delays are added like those needed to achieve the linear combination or the delay suffered by a packet to wait the corresponding packets arriving from other links. The objective of our work is to integrate these different parameters to establish under which conditions, network coding can decrease worst case end-to-end delays compared to a classical routing/multiplexing approach.

Two network coding strategies are described here. They are based on fixed network code determined *a priori*. To ensure that receivers decode all the received packets, the concept of packet generation introduced in [9] is used. The first strategy performs a combination of all the packets of the same generation in each coding node. The second strategy, called *fast forwarding strategy*, allows partial combination in intermediate coding nodes and uses packet checksums at the receiver to check the recovery of the initial packets.

After a short introduction of the main concepts of network calculus, the network hypotheses are detailed in Section III. The two network coding strategies are presented in Section IV and their QoS guarantees are evaluated. A similar analysis is provided in Section V for a classical multiplexing approach. In Section VI, these different approaches are compared in a study case. Finally, Section VII concludes.

## II. Network Calculus

Network Calculus is a framework providing deterministic bounds to end-to-end delays, backlogs and other QoS parameters by using the Min-Plus algebra. This theory was introduced and developed by Le Boudec and Thiran in [1] by generalizing previous works such as [10][11]. The following definitions and results are extracted from [1] where a detailed presentation of this theory can be found.

1) A data stream $F$ transmitted on a link can be described by the *cumulative function* $R(t)$, such that for any $y > x$, $R(y) - R(x)$ is the quantity of data transmitted on this link during the time interval $[x, y]$.
2) Let $F$ be a data stream with cumulative function $R(t)$. We say that an increasing function $\alpha$ is an arrival curve of $F$ (or equivalently $R$) if for any $0 \leq t_1 \leq t_2$, $R(t_2) - R(t_1) \leq \alpha(t_2 - t_1)$. A common class of arrival curves are the affine functions $\gamma_{r,b}(t) = rt + b$ for $t > 0$ and 0 otherwise. The curve $\gamma_{r,b}(t)$ represents the arrival curve of the leaky bucket controller with leak rate $r$ and bucket size $b$.

978-1-4244-2563-1/08/$25.00 ©2008 IEEE

3) The min-plus convolution of two functions $X$ and $Y$ is defined as $X(t) \otimes Y(t) = inf_{0 \leq s \leq t}(X(s) + Y(t-s))$. It can be shown that $\alpha$ is an arrival curve of $R$ if and only if $R \leq R \otimes \alpha$.
4) Let $R^{out}$ be the output flow of a node with one input flow $R$. We say that the node offers a service curve $\beta(t)$ to $R$ if for any $t > 0$, $R^{out}(t) \geq R(t) \otimes \beta(t)$.
5) Assume that a flow $R(t)$, constrained by an arrival curve $\alpha(t)$ traverses a system offering a service curve of $\beta$. The output flow $R^{out}$ is constrained by the arrival curve $\alpha \oslash \beta$, where $(\alpha \oslash \beta)(t) = sup_{v \geq 0}\{\alpha(t+v) - \beta(v)\}$ .
6) The Burst Delay Service curve $\delta_T$ is equal to $\infty$ if $t > T$ and 0 else.
7) The rate latency service curve $\beta_{R,T} = R[t-T]^+$ is equal to $R(t-T)$ if $t > T$ and 0 else.
8) The backlog of a flow $R$ in the node at the time $t$ is the amount of data "in transit" in the node. This backlog, defined as $R(t) - R^{out}(t)$ for all t, satisfies $R(t) - R^{out}(t) \leq sup_{s>0}\{\alpha(s) - \beta(s)\}$.

## III. Network Hypotheses

Consider a communication network represented by an acyclic directed graph $G = (V, E)$ with a vertex set $V = \{v_1, \ldots, v_m\}$ and an edge set $E$. The directed edge connecting the node $v_i$ to the node $v_j$ is denoted by $e_{i,j}$.

We assume that all the nodes are synchronized. Each edge $e_{i,j}$ has a capacity $C_{i,j}$ $(bits/sec)$, meaning that a packet of $L$ bits is transmitted in at least $L/C_{i,j}$ seconds. Since the system is assumed to provide QoS guarantees, we consider that, for each edge $e_{i,j}$, the maximum transmission delay of a packet of $L$ bits is known and equal to $L/C_{i,j} + T_{i,j} = \omega_{i,j} + T_{i,j}$. In other words, the edge $e_{i,j}$ has the rate latency service curve $\beta_{C_{i,j},T_{i,j}}(t)$. We suppose that the capacity of every output edge of a node is greater or equal than the sum of capacities of all input edges. This hypothesis is used to be fair with the routing approach, but for network coding, it is sufficient to have the output capacity greater than the maximum of the input capacities.

Nodes are divided into three categories. *Source nodes* generate independent unicast or multicast input flows. *Intermediate nodes* are the nodes situated between the sources and the receiver nodes. They able to perform network coding operations following a given strategy. We assume that the time necessary to perform a linear combination of several packets does not depend on the number of packets and is upper-bounded by $T_{lc}$. We assume that a linear network code was determined *a priori* for the considered network. Consequently, each node knows how to combine the input flows to produce output flows. The other nodes are the *receiver nodes* which receive and decode the combined flows. They have the same properties than intermediate nodes.

The flows generated by the source are composed of packets of fixed length $L$. They verify two constraints. First, we assume that at most one packet is generated in each time interval $[t_i, t_i + \Delta[$. Each time interval is labeled with a generation number carried in the packet header. Following [9], the linear network code only combines packets belonging to the same generation, even if packets from different generations are simultaneously present in a node. This constraint implies that the maximum rate of the flow is $L/\Delta$. The second constraint imposed to the flows is to verify bounds on burstiness and throughput. This constraint is represented by an affine arrival curves (see point 2- in Section II).

## IV. Network coding strategies for networks providing QoS guarantees

### A. General Strategy

This strategy is based on the classical definition of the network coding. Let us consider an intermediate node with $n$ input flows and one output flow (see Figure 1). We consider that for each generation $i$, a deadline of the arrival time of $P_i$ is known. With the network hypotheses presented in Section III, the linear combination corresponding to a generation $i$ is done as soon as, for all the input flows, at least one of the following points is verified :

- all the packets of the generation $i$ is in the buffers.
- some packets of the generation $i$ are not in the buffers and the deadline of the arrival time of the generation $i$ is exceeded or their corresponding packet of the generation $i+1$ is in the buffer.

The last point indicates that the corresponding flow does not contain a packet of the generation $i$. In this case, the linear combination is only done with the packets of the generation $i$ present in the node. Algebraically, this is equivalent to replacing the missing packets by packets full of zeros.

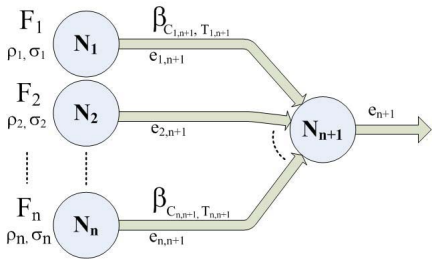


Fig. 1. Node with $n$ input flows

*1) Delay analysis:* The delays suffered by the packets in a node have several reasons. First, to perform a linear combination of several packets, the node must wait all the packets. Since the network provides delays guarantees, a node is able to compute a time limit for reception of a given generation. The maximum time spent by a packet of the flow $i$ in the node $n+1$ to wait the packets of the same generation arriving from other links is denoted by $T_{B_i}^{n+1}$. We represent this delay with the service curve $\beta_{B_i}^{n+1}(t) = \delta_{T_{B_i}^{n+1}}(t)$. If a packet of a given generation is not received before the time limit, the node assumes that the source did not generate a packet in this time interval, and it performs combinations with the available packets. Algebraically, this is equivalent to replacing the missing packets by packets full of zeros.

Second, linear combination of the packets adds a delay upperbounded by $T_{lc}$. The service curve associated to this operation is $\delta_{T_{lc}}$. Finally, the processing of an encoded packet by the intermediate node $n+1$ to transmit it is represented by the rate latency service curve $\beta_{C_{n+1},\tau_{n+1}}(t) = C_{n+1}(t-\tau_{n+1})$ where $C_{n+1}$ is the capacity of the output link. We denote by $\beta^i_{n+1}(t)$ the total service curve provided by the node to combine and transmit the packet belonging to the flow $i$. Since the service curve of an intermediate node is the convolution of the service curves offered by its elements, the service curve offered by an intermediate node $n+1$ to a flow $i$ is therefore:

$$\begin{aligned}\beta^i_{n+1}(t) &= \beta_{C_{n+1},\tau_{n+1}}(t) \otimes \delta_{T^{n+1}_{B_i}} \otimes \delta_{T_{lc}}(t) \\ &= \beta_{C_{n+1},\tau_{n+1}+T^{n+1}_{B_i}+T_{lc}}\end{aligned}$$

The service provided by the receiver nodes is equal to the one provided by the intermediate nodes. Indeed, the decoding of the network code is also a linear combination of input packets and thus, the problem is the same that for the intermediate nodes.

*2) Maximum delay at an intermediate node :* Let us now consider the Figure 1. Each flow $F_i$, $i = 1, \ldots, n$, is constrained by $\gamma_{\rho_i,\sigma_i}$. These flows are transmitted over the edges $\{e_{1,n+1}, \ldots, e_{n,n+1}\}$ to an intermediate node $N_{n+1}$ in order to be combined. The service curve offered by an edge $e_{i,n+1}$ is $\beta_{C_{i,n+1},T_{i,n+1}}$. Let us consider a packet generated in the time interval $\{t, t+\Delta\}$. Let us assume that the transmission delay on the edge $e_{i,3}$ is in the range $[\omega_{i,n+1}, T_{i,n+1}+\omega_{i,n+1}]$. It follows that the earliest arrival time of this packet in the node is $t+\omega_{i,n+1}$ and the latest arrival time is $t+\Delta+T_{i,n+1}+\omega_{i,n+1}$. The maximum waiting time of a packet of flow $F_i$ in the receiving buffer $B_i$ is

$$T^{n+1}_{B_i} = \Delta + \max_{\substack{j=1\ldots,n \\ j\neq i}} \{T_{j,n+1} + \omega_{j,n+1}\} - \omega_{i,n+1}$$

The associated service curve is $\beta^{n+1}_{B_i}(t) = \delta^{n+1}_{T_{B_i}}(t)$. The processings of the linear combination and the transmission offer respectively the service curves $\beta_{lc}(t) = \delta_{T_{lc}}(t)$ and $\beta_{C_{n+1},\tau_{n+1}}(t) = C_{n+1}(t-\tau_{n+1})$ where $C_{n+1}$ is the capacity of $e_{n+1}$. Suppose that $C_{i,n+1} \leq C_{n+1}$ and $\rho_i \leq C_{i,n+1}$ for $i = 1, 2, \ldots, n$.

The total service curve offered to the flow $F_i$ by the coding node $n+1$ is $\beta^i_{n+1}(t) = \beta_{C_{n+1},T^*_i}$, where $T^*_i = T^{n+1}_{B_i} + T_{lc} + \tau_{n+1}$. The maximum delay experienced by a packet of $F_i$ in this node is

$$T^{cn}_i = T^{n+1}_{B_i} + T_{lc} + \tau_{n+1} + \frac{\sigma_i + \rho_i T_{i,n+1}}{C_{n+1}} \quad (1)$$

An arrival curve of a subflow corresponding to $F_i$ at the output of this intermediate node is $\gamma_{\rho_i,\sigma^*_i}$ where $\sigma^*_i = \sigma_i + \rho_i T_{i,n+1} + \rho_i(T^{n+1}_{B_i} + T_{lc} + \tau_{n+1})$. So the necessary buffer size for the flow $F_i$ in the coding node $n+1$ is greater than or equal to $\sigma^*_i$.

### *B. Fast forwarding strategy*

*1) Description of the system and hypotheses:* The system presented in Section IV-A was designed to work with a given number of flows and is optimal when all the flows are active. When some of the flows are idle, the others flows wait them in the coding nodes and consequently, their end-to-end delays are increased. The improvement we propose allows to avoid this problem by authorizing the packets to leave the coding node even if the whole generation is not arrived. This strategy is called *fast forwarding*.

The network hypotheses are the same than in Section IV-A. All the sources are synchronized and in each period of time $\Delta$, at most one packet is generated by each source. All packets generated during the same period of time carries the same generation number. The delay experienced by a packet on a link and the throughput of a link are variable but we assume that each link $e_{i,j}$ provides a guaranteed service represented by the service curve $\beta_{C_{i,j},T_{i,j}}(t)$.

In addition to the hypotheses done in the previous system, we consider that each packet contains a checksum computed on the whole packet payload. We propose to use a classical checksum used in higher layers of IP protocols, such as *e. g.* the one of UDP [12]. Since it is defined as the "16-bit one's complement of the one's complement sum of" the data, it is based on the sums of 16-bit integers (modulo $2^{16}$), and thus it is not linear in any subfield of the field used by the network code (we assume that this field has the form $\mathbb{F}_{2^m}$). It follows that the linear combination (by the network code) of the checksums of two packets is not equal to the checksum of the same linear combination of the two packets (with a high probability). Note that contrary to the checksum, the CRC (cyclic redundancy check - see e.g. [13]) are linear over the field $\mathbb{F}_2$. Therefore, since $\mathbb{F}_2 \subset \mathbb{F}_{2^m}$, some linear combinations in $\mathbb{F}_{2^m}$ are also linear combinations in $\mathbb{F}_2$. The non-linear property of the checksum is used by the receiver to check the validity of a current decoding.

The structure of the intermediate node is also modified. The main difference is that the nodes contain only one buffer. The management of this buffer with the new strategy is described in the next paragraph.

*2) Fast forwarding strategy at the intermediate nodes:* Let us consider an intermediate node with $n$ input flows and one output flow (see Figure 2) with the network hypotheses presented in Sections III and IV-B.1. Suppose that a packet of a given generation $X$ arrives at the coding node $n+1$ at time $t$. The fast forwarding strategy of this coding node is the following :

- If the buffer is empty, the packet is multiplied by the finite field coefficient determined by the network code and is transmitted over the output link (if this link is not used by another packet transmission started before time $t$).
- If the buffer is not empty:
  - if there is not a packet of the generation $X$ in the buffer, the packet is multiplied by its corresponding finite field coefficient and added at the end of the buffer. For example, on Figure 2, the packet $P^1_3$ arriving from node $N_1$ is added at the end of the buffer.

– if there is a packet of the generation $X$ in the buffer, the arriving packet is multiplied by its corresponding finite field coefficient and is directly summed to the packet of the its generation in the buffer. For example, on Figure 2, the packet $P_5^2$ arriving from node $N_2$ is summed to the packet $P_5^1$ already present in the buffer.

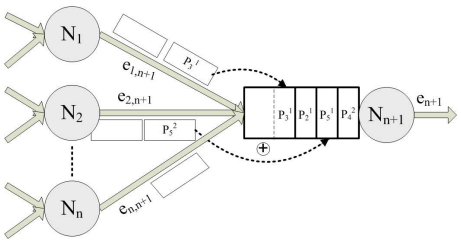


Fig. 2. Fast forwarding strategy

Note that, this strategy could lead to generation desequencing (like in the Figure 2).

To estimate the end-to-end delays and the buffer size, we must determine the maximum delay suffered by a packet in an intermediate node. From the strategy described previously, it can be deduced that a packet must wait at most the time needed to transmit the maximal number of different generations which can be found simultaneously in the intermediate node (when the packet arrives at the node). The arrival time at each intermediate node and the intergeneration times are used to calculate this number.

*a) Arrival time at an intermediate node:* Each generation has a predetermined minimum and maximum arrival times at each node. Each node can evaluate these arrival times for a packet of a given generation by asking the father nodes the minimum and maximum sending times and by considering the additional transmission delays (given by the properties and the service curves of the links). Consequently, it can calculate the minimum and maximum arrival times for a generation by finding the minimum and the maximum arrival times between the minimum and maximum arrival times of all packets belonging to the this generation.

For example, let us consider the node in Figure 2 as first-order intermediate node (e.g. it gets its input data directly from sources ). The arrival time of a generation $X$ which is generated in an interval $[t, t+\Delta]$ at the input of this intermediate node $n+1$ will be bounded by:

$$\begin{bmatrix} t+\omega_{1,n+1} & , & t+\Delta+T_{1,n+1}+\omega_{1,n+1} \\ t+\omega_{2,n+1} & , & t+\Delta+T_{2,n+1}+\omega_{2,n+1} \\ \vdots & \vdots & \vdots \\ \underbrace{t+\omega_{n,n+1}}_{min} & , & \underbrace{t+\Delta+T_{n,n+1}+\omega_{n,n+1}}_{max} \end{bmatrix}$$

$$t_X = [t+\min\{\omega_{i,n+1}\}, t+\Delta+\max\{T_{i,n+1}+\omega_{i,n+1}\}]$$

$$[t_X]_{min} = t+\min\{\omega_{i,n+1}\}$$

$$[t_X]_{max} = t+\Delta+\max\{T_{i,n+1}+\omega_{i,n+1}\}$$

where $i = 1, 2, \cdots, n$.

Now suppose that the node $n+1$ is an intermediate node inside of network. The arrival time of a generation $X$ which is generated in an interval $[t, t+\Delta]$ at the input of this node will be bounded by:

$$t_X = \quad [[t_X]_{min}, [t_X]_{max}]$$

where $[t_X]_{min} = t+\min\{\omega_{i,n+1}+\left[A^X_{i,n+1}\right]_{min}\}$ and $[t_X]_{max} = t+\Delta+\max\{T_{i,n+1}+\omega_{i,n+1}+\left[A^X_{i,n+1}\right]_{max}\}$ for $i = 1, 2, \cdots, n$, $\left[A^X_{i,n+1}\right]_{min}$ is the minimum delay suffered by a packet of generation $X$ from its source to the input of edge $e_{i,n+1}$ and $\left[A^X_{i,n+1}\right]_{max}$ is the maximum delay suffered by a packet of generation $X$ from its source to the input of edge $e_{i,n+1}$.

*b) Time separating generations at an intermediate node:* We can calculate the time separating two consecutive generations $\{X-1, X\}$ by exploiting the previous results. For a generation $X$ the minimum arrival time at a node $n+1$ on a edge $e_{i,n+1}$ (see Figure 2) is $t+\min\{\omega_{i,n+1}+\left[A^X_{i,n+1}\right]_{min}\}$. On the same edge, the minimum arrival time of generation $X-1$ will be $t-\Delta+\min\{\omega_{i,n+1}+\left[A^X_{i,n+1}\right]_{min}\}$. We can note that the time which separate the minimum arrival times of two consecutive generations is equal to $\Delta$. Now the maximum arrival time of generation $X$ at this node $n+1$ is $t+\Delta+\max\{T_{i,n+1}+\omega_{i,n+1}+\left[A^X_{i,n+1}\right]_{max}\}$ surly on edge $e_{i,n+1}$, while for the generation $X-1$ is $t+\max\{T_{i,n+1}+\omega_{i,n+1}+\left[A^X_{i,n+1}\right]_{max}\}$. We find the same result that the time separating both maximum arrival time of generations equal to $\Delta$.

*c) Buffer size at an intermediate node:* The maximum number of generations present in the buffer when a packet of the generation $X$ arrives at the node can be deduced from the maximum arrival time and the time separating generations. The maximum number of generations will be $\{[t_X]_{max}-[t_X]_{min}\}/\Delta$. The maximum buffer size of an intermediate node will be $\max\{[t_X]_{max}-[t_X]_{min}\}L/\Delta$ for $X = 1, 2, \cdots$. Then the maximum delay suffered by a packet of generation $X$ in the buffer of a node $n+1$ is

$$T_B^{n+1} = \frac{([t_X]_{max}-[t_X]_{min})L/\Delta}{C_{n+1}}$$

*d) Maximum delay at an intermediate node:* The maximum delay of a packet $X$ of a flow $F_i$, constrained by $\gamma_{\rho_i,\sigma_i}(t)$, at an intermediate node $n+1$ is equal to

$$T_i^{cn} = T_{B_i}^{n+1} + T_{lc} + \tau_{n+1} + \frac{\sigma_i + \rho_i T_{i,n+1}}{C_{n+1}}$$

*3) Fast forwarding strategy at the receiver nodes:* These modifications of the coding node strategy does not affect the decoding process. Let us illustrate it on a simple example.

Let us consider the case where three flows are combined in a node to produce an output flow. For a given generation $X$, the packets $p_X^1, p_X^2$ and $p_X^3$ (respectively from the first, second and third flows) have to be combined to produce the

packet $p_X^{out} = \alpha_1 p_X^1 + \alpha_2 p_X^2 + \alpha_3 p_X^3$ where $\alpha_i$ are finite field coefficients. Assume that $p_X^1$ and $p_X^2$ simultaneously arrive in the node. Since the third flow can be idle, they are authorized to leave the node under the form $p_X^{out,1} = \alpha_1 p_X^1 + \alpha_2 p_X^2$. If the packet $p_X^3$ arrives at the node after $p_X^{out,1}$ was transmitted, it is "encoded" and leaves the node under the form $p_X^{out,2} = \alpha_3 p_X^3$. In a following node in the path(s) toward the receiver, $p_X^{out,2}$ can catch up with $p_X^{out,1}$ and then, $p_X^{out,1}$ and $p_X^{out,2}$ are summed in this node.

If this is not the case, *i. e.* $p_X^{out,1}$ and $p_X^{out,2}$ arrive separately at the receiver node, the receiver must be able to check if $p_X^{out,1}$ is the only packet arriving from this path (i. e. the source generating $p_3$ did not produce a packet in the generation $X$) or if $p_X^{out,2}$ will arrive.

The decoding of a generation begins as soon as the first packet of this generation arrives and it is not necessary to wait the arrival of all packets of this generation. For that, it begins the decoding of the generation $X$ with the present packets. Note that this decoding is simply a matrix-vector multiplication, where the matrix corresponds to the inverse matrix of the network code and the vector is the set of received packets (see e. g. [14]). The receiver verifies then the checksum of the obtained packets. Since the checksums are not linear (in the finite field of the network code), a packet with a correct checksum is necessarily (or with a high probability) a packet generated by a source (see Section ). If the checksum is not correct, this means that at least one of the input packets is not correct and then, the receiver has to wait additional packets of the generation $X$. When a new packet arrives, it updates the obtained packets and verifies again the checksums. Note that this update does not need a complete matrix vector multiplication, but simply a scalar multiplication of the corresponding row of the matrix and an addition of two vectors.

Since we assume that no packet is lost nor erroneous, then, when the last packet arrives, the initial packets are recovered.

## V. Classical routing/multiplexing strategy

Let us consider the example represented on Figure 1. We suppose that $n$ source nodes $N_1, \ldots, N_n$ generate $n$ flows $F_1, \ldots, F_n$. These flows are respectively constrained by the affine arrival curves $\gamma_{\rho_1,\sigma_1}, \ldots, \gamma_{\rho_n,\sigma_n}$. These flows are multiplexed by the node $N_{n+1}$ and transmitted over the edge $e_{n+1}$. The service curves offered by the edges $e_{1,n+1}, \ldots, e_{n,n+1}$ are respectively $\beta_{C_{1,n+1},T_{1,n+1}}, \ldots, \beta_{C_{n,n+1},T_{n,n+1}}$. We assume that $\rho_i < C_{i,n+1}$. Following II-5 and for $i = 1, \ldots, n$, at the output of edge $e_{i,n+1}$, flow $F_i$ has the arrival curve $\gamma_{\rho_i,\sigma_i} \oslash \beta_{C_{i,n+1},T_{i,n+1}} = \gamma_{\rho_i,\sigma_i+\rho_i T_{i,n+1}}$. We suppose that the FIFO multiplexer node $N_{n+1}$ offers a service curve $\beta_{C_{n+1},\tau_{n+1}}$ to the aggregated flow and that $\sum_{i=1}^{n} \rho_i < C_{n+1}$.

By generalizing [1, Theorem 6.2.3], it can be shown that the FIFO multiplexer node $N_{n+1}$ offers to the $k^{th}$ flow the service curve

$$\begin{aligned} \beta_k(t) &= \beta_{C_{n+1},\tau_{n+1}}(t) - \sum_{i=1,i\neq k}^{n} \gamma_{\rho_i,\sigma_i+\rho_i T_{i,n+1}}(t) \\ &= \beta_{(C_{n+1}-\sum_{i=1,i\neq k}^{n}\rho_i),T_k^*} \end{aligned} \quad (2)$$

where $T_k^* = \tau_{n+1} + \frac{\sum_{i=1,i\neq k}^{n}(\sigma_i+\rho_i T_{i,n+1})}{C_{n+1}}$.

The maximum delay of a flow $F_k$ at a multiplexing node is

$$T_k^{mn} = \tau_{n+1} + \frac{\sum_{i=1,i\neq k}^{n}(\sigma_i+\rho_i T_{i,n+1})}{C_{n+1}} + \frac{(\sigma_k+\rho_k T_{k,n+1})}{C_{n+1}-\sum_{i=1,i\neq k}^{n}\rho_i}$$

Then the arrival curve of the $k^{th}$ flow at the output of the multiplexer is equal to $\gamma_{\rho_k,\sigma_k^*}$ where

$$\sigma_k^* = \sigma_k + \rho_k(T_{k,n+1} + \tau_{n+1} + \frac{\sum_{i=1,i\neq k}^{n}(\sigma_i+\rho_i T_{i,n+1})}{C_{n+1}})$$

## VI. Case Study

Let us consider the network presented in Figure 3. In this

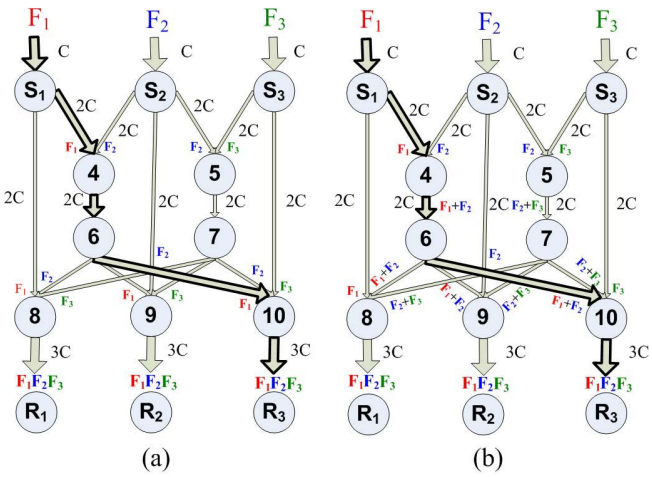


Fig. 3. Interest of Network Coding for QoS parameters

case, sources $S_1$, $S_2$ and $S_3$ multicast the flows $F_1$, $F_2$ and $F_3$ towards three receivers $R_1$,$R_2$ and $R_3$. $F_1$, $F_2$ and $F_3$ are constrained by the same affine arrival curve $\gamma_{C,\sigma}(t)$. All links have capacity $2C$ except the input links of the receivers which have capacity $3C$. It must be noted that under these hypotheses, network coding does not improve the throughput compared to the multiplexing approach.

The service delay over the links $e_{i,j}$ is $T_{i,j} = T$. Then each link $e_{i,j}$ provides a service curve $\beta_{2C,T}(t)$ or $\beta_{3C,T}(t)$. Each multiplexer node $N_k$ offers a service curve $\beta_{2C,\tau_k}(t)$ where $\tau_k$ is the service delay to aggregate flow. Each intermediate node $N_k$ offers a service curve $\beta_{2C,T_B^k+T_{lc}+\tau_k}(t) = 2C(t - T_B^k -$

$T_{lc} - \tau_k$) where $T_B^k$ is the maximum time spent by a packet in the buffers while waiting for corresponding packets of others flows. $T_{lc}$ denotes the maximum time needed to achieve a linear combination of packets and $\tau_k$ is the service delay to transmit a packet.

With the conditions described previously, the worst case delay for multiplexing and coding cases is obtained on the paths crossing the maximum of nodes, i.e. for paths crossing 3 nodes. Since all paths, with 3 nodes, have the same property, we can choose one of them and study its worst case delay. We choose the path from $S_1$ to $R_3$ which cross nodes 4, 6 and 10.

In the *multiplexing strategy*, the maximum delay of flow $F_1$ at the output of multiplexer 10 can be obtained by simply applying the formula given in previous sections. The maximum delay is:

$$T^{mn}_{S_1,R_3} = 5T + \frac{11}{3}\tau + \frac{7\sigma}{3C} \quad (3)$$

In the *general coding strategy*, with the results presented in the previous sections, the worst case delay is equal to:

$$T^{nc}_{S_1,R_3} = 3T + 3\tau + 2T_{lc} + \Delta\{2 + 4\frac{T}{\Delta}\} + \frac{\sigma}{2C} \quad (4)$$

The maximum backlog of flow $F_1$ at the output of intermediate node 10 is:

$$\begin{aligned}\sigma^{nc}_{S_1,R_3} &= \sigma_1 + \rho_1 T_{tot} \\ &= \sigma + C(3T + 3\tau + 2T_{lc} + \Delta\{2 + 4\tfrac{T}{\Delta}\})\end{aligned}$$

with the *fast forwarding strategy*, the worst case delay is equal to:

$$T^{nc}_{S_1,R_3} = 3T + 3\tau + 2T_{lc} + \frac{L}{C}\{\frac{5}{6} + \frac{3}{2}\frac{T}{\Delta}\} + \frac{\sigma}{2C} \quad (5)$$

while the maximum backlog at the output of intermediate node 10 is:

$$\begin{aligned}\sigma^{nc}_{S_1,R_3} &= \sigma_1 + \rho_1 T_{tot} \\ &= \sigma + C(3T + 3\tau + 2T_{lc} + \tfrac{L}{C}\{\tfrac{5}{6} + \tfrac{3}{2}\tfrac{T}{\Delta}\})\end{aligned}$$

The comparison of the worst case delays of the two coding strategies (*i.e.* Equations 4 and 5) directly shows that fast forwarding strategy obtains better end-to-end delay bounds than general coding strategy.

The comparison between *multiplexing strategy* and *fast forwarding strategy* can be done from Equations 3 and 5. The end-to-end delay bound obtained with *fast forwarding strategy* is better than with *multiplexing strategy* if and only if:

$$T_{lc} \leq \frac{\tau}{3} + T(1 - \frac{3.L}{4.C.\Delta}) + \frac{11.\sigma}{12.C} - \frac{5.L}{12.C} \quad (6)$$

The conclusion of this comparison depends on the relationships between the different parameters. Unsurprisingly, the performance of network coding strongly depends on the value of $T_{lc}$, which is the delay due to a combination of two packets. For a fixed $T_{lc}$, the interest of network coding grows when the parameters $\tau$ and $T$ are increased. These parameters are respectively the service delay of a node and the transmission delay on a link. Note the coefficient of $T$ is strictly greater than 0 because the time needed to send a packet ($L/C$) is necessarily lower than $\Delta$ which is the duration of a generation range. It can also be observed (with the parameter $\sigma$) that the more the traffic is bursty, the more network coding is better.

## VII. Conclusion

This paper has presented two network coding strategies for networks providing QoS guarantees. These two strategies are evaluated in terms of maximum delays for a packet to be treated by a node. In a study case, where the network coding strategy didn't improve the throughput performance, this delay analysis was generalized at the network level and compared to a classical multiplexing strategy. We have presented the relationship between different parameters (such as coding delay, transmission delay, throughput, burstiness, generation duration, ...) in order to determine in which conditions the network coding allows to decrease end-to-end delays guarantees.


## Acknowledgements

The authors wish to thanks Emmanuel Lochin for his help.


## References


[1] J.-Y. L. Boudec and P. Thiran, *Network Calculus A Theory of Deterministic Queuing Systems for the Internet*, ser. Series: Lecture Notes in Computer Science, Vol. 2050. Springer Verlag, 2001.

[2] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flows," *IEEE Transactions on Information Theory*, vol. 46, pp. 1204–1216, July 2000.

[3] T. Ho, R. Koetter, M. Medard, D. Karger, and M. Effros, "The benefits of coding over routing in a randomized setting," *International Symposium on Information Theory (ISIT),*, 2003.

[4] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, and J. Crowcroft, "Xors in the air: practical wireless network coding," in *SIGCOMM '06: Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*, 2006, pp. 243–254.

[5] Y. Zhu, B. Li, and J. Guo, "Multicast with network coding in application-layer overlay networks," *IEEE J. Selected Areas in Comm.*, vol. 22, no. 1, January 2004.

[6] Z. Li, B. Li, D. Jiang, and L. C. Lau, "On achieving optimal throughput with network coding," *In Proc. of IEEE INFOCOM 2005*, March 2005.

[7] Y. Ma, W. Li, P. Fan, and X. Liu, "Queuing model and delay analysis on network coding," in *ISCIT*, October 2005.

[8] A. Eryilmaz, A. Ozdaglar, and M. Medard, "On delay performance gains from network coding," in *In Conference on Information Sciences and Systems (CISS)*, March 2006.

[9] P. A. Chou, Y. Wu, and K. Jain, "Practical network coding," *41st Allerton Conference*, Oct. 2003.

[10] R. L. Cruz, "A calculus for network delay, part i : Network elements in isolation," *IEEE Transactions on Information Theory*, vol. 37, no. 1, pp. 114–131, January 1991.

[11] ——, "A calculus for network delay, part ii : Network analysis," *IEEE Transactions on Information Theory*, vol. 37, no. 1, pp. 132 – 141, January 1991.

[12] J. Postel, "(RFC 768) User Datagram Protocol," Aug. 1980.

[13] J. Stone and al., "RFC 3309: Stream control transmission protocol (sctp) checksum change," 2002.

[14] R. Koetter and M. Médard, "An algebraic approach to network coding," *IEEE/ACM Transactions on Networking*, vol. 11, no. 5, pp. 782–795, October 2003.