Université catholique de Louvain
Faculté des Sciences Appliquées
CENTER FOR OPERATIONS RESEARCH
AND
ECONOMETRICS
and
LABORATOIRE DE TÉLÉCOMMUNICATIONS
ET
TÉLÉDÉTECTION

B - 1348 Louvain-la-Neuve Belgique

Optimization of Multimedia Flows over Data Networks : the Core Location Problem and the Peakedness Characterization

Jean-François Macq

Thèse présentée en vue de l'obtention du grade de
Docteur en Sciences Appliquées

Examination Committee:

Benoît MACQ (UCL/TELE) - *Supervisor*
Laurence WOLSEY (UCL/OPT and CORE) - *Supervisor*
Philippe CHEVALIER (UCL/POMS and CORE)
Bernard FORTZ (UCL/POMS)
Michel GOEMANS (Massachusetts Institute of Technology, U.S.A.)
Nico VANDAELE (Universiteit Antwerpen and UCL/CORE)
Jean-Didier LEGAT (UCL/DICE) - *President*

May 2005

Acknowledgements

First of all, I wish to thank my supervisors, Benoit Macq and Laurence Wolsey, for their guidance and continuous support during these four years. I am very grateful not only for their technical expertise but also for their ability to make me self-confident in moments of doubt. Moreover, working on a subject at the frontier of their respective research areas has been a very rewarding experience.

The results presented in Chapter 3 were obtained in collaboration with Michel Goemans during my stay at the Massachusetts Institute of Technology. I wish to express him my deep gratitude for having offered me the opportunity to spend a wonderful year in Boston.

I also owe many of the ideas presented in this thesis to fruitful discussions with colleagues at CORE and at the TELE lab. I am in particular very grateful to Philippe Chevalier who partly supervised the work described in the second part of this thesis. Many thanks also to the members of the examination committee for their numerous and helpful comments which allowed me to improve the manuscript.

When doing full-time research, friendship is certainly an indispensable ingredient to preserve one's mental health. Besides the old good friends, many colleagues have intensely contributed to create a very friendly and relaxed working atmosphere.

Last but not least, I thank my parents and family for their everlasting love and support. I can hardly say how thankful I am to Isa for her tremendous support and patience throughout these four years and more particularly during the last few months. This thesis is dedicated to them.

The research reported in this thesis has been funded by the Belgian National Fund for Scientific Research (F.N.R.S.) and partly by the Belgian American Educational Foundation (B.A.E.F.).

Contents

List of Abbreviations and Notations

ARQ	Automatic Repeat reQuest
CDN	Content Delivery Network
CBT	Core Based Tree
$\mathsf{Cov}[X, Y]$	Covariance of random variables X and Y
$\mathsf{E}[X]$	Expectation of random variable X
FEC	Forward Error Correction
γ_{XY}	mutual variability of processes X and Y
iid	independent and identically distributed
IP	Internet Protocol
$k_X(t)$	Covariance density function of process X
$k_{XY}(t)$	Mutual covariance density function of processes X and Y
λ_X	Arrival rate of process X
MMFP	Markov Modulated Fluid Process
MBCCL	Minimum Bandwidth Cost Core Location
$N_X(t)$	Counting process associated with process X
$P(R, K)$	Probability of irrecoverable loss for a $(R, K)-$FEC fluid flow
π_i	Stationary probability of state $i \in \{0, 1\}$ for a (μ_0, μ_1)-Markov process
PIL	Probability of irrecoverable loss
$P_{IL}(R, K)$	Probability of irrecoverable loss of a $(R + K, K)$-FEC block
PDF	Probability Density Function
$\mathsf{P}[A]$	Probability of event A
$\mathsf{P}[A\|B]$	Probability of event A conditioned to event B
TCP	Transmission Control Protocol
$\mathsf{Var}[X]$	Variance of random variable X

List of Figures

Chapter 1

Introduction

Over the past years, the widespread penetration of broadband access to the Internet has begun to provide home users with an ever growing number of multimedia applications. However, whereas the Internet can nowadays support telephony and low resolution video transmissions with sufficient quality, it does not yet meet the requirements for more demanding applications such as high quality Video on Demand or services similar to TV broadcasting. The need for higher rate transmission technologies is not the only explanation of the mismatch between the Internet and the requirements of many multimedia applications. More fundamental reasons may be found in a few key decisions that were made in the early 1970s when designing the architecture of *the network of networks* ([10],[13]).

1.1 Design Philosophy of the Internet

The goal was then to devise a uniform communication interface between various types of networks. In order to obtain a flexible and scalable architecture, the *Internet Protocol (IP)* has only been provided with the following minimal functionalities ([34]) :

1. *addressing*, i.e. a way to unequivocally identify any destination in the network,
2. *routing*, which defines how intermediate nodes in the network, called *routers*, forward data packets towards their final destination.
3. *the IP packet format* which guarantees that information relevant to the previous functionalities is mentioned within each data packet in a uniform way.

Routing is arguably the most complex of these functionalities. It is generally achieved by maintaining at each router a *routing table* which, given the address of the final destination of a packet to be processed, indicates the next router to which the packet must be forwarded. A *routing protocol* specifies how these routing tables are constructed. Most of the time, the protocol functionalities include a distributed algorithm which computes shortest paths from the router to the other nodes in the network. This underlying optimization is based on classical algorithms such as the Dijkstra ([17]) or Bellman-Ford ([6]) algorithms. The length of these paths may be measured as the number of links used ("hop-count" metric) or accordingly to a given metric like average link delays or costs set by network operators. Note that the protocol must ensure that routing data are kept consistent throughout all routers when the network metric or topology change. Routing complexity typically lies in these update mechanisms.

IP flexibility is to a large extent achieved by the fact that the protocol does not assume any transmission reliability from the underlying communication technologies. But that flexibility comes at a price : in return, IP itself does not guarantee the delivery of packets to their destination, nor packet data integrity during their transmission. This lazy behavior of IP is often referred as the *Best Effort* service. More generally, the Internet philosophy has mainly been dictated by the *end-to-end argument* ([52]) : mechanisms inside the network should be simple and all the complicated machinery (including transmission reliability, flow control, security,...) be moved to the edge of the network, where it can be implemented accordingly to the end-hosts requirements. In particular, the task of error detection and correction is left to a *transport protocol* operating above IP. In the Internet, the *Transmission Control Protocol (TCP)* has been chosen as the standard transport protocol. It achieves a reliable packet transmission by using an *Automatic Repeat reQuest (ARQ)* mechanism : upon detection of packet losses or data corruption, TCP explicitly requests a retransmission of affected packets.

The dramatic growth of the Internet since its creation more than 30 years ago gives us evidence that all these mechanisms have been successful in providing its users with classical services such as e-mail, web browsing, file transfer, remote computer access,... In the next section, we attempt to explain why the Internet architecture is not suited to some multimedia applications

1.2 Shortcomings of the Internet Architecture for Multimedia Application Support

We describe here two limitations of the Internet architecture with respect to some multimedia applications. The reason why we emphasize these particular issues is that they are directly related to the problems we address in this thesis.

1.2.1 Unicast vs Multicast Routing

Native IP routing mechanisms mentioned in Section 1.1 were initially designed for *unicast* routing, that is for routing packets from a source to a single destination. However, for several multimedia applications, the same data packets have to be sent simultaneously to several destinations. Consider as an example the streaming of a live video. In such a situation, using independent unicast routes between the source and each user usually leads to a waste of network resources as these routes may partially overlap. Sending several times the same data on a link is indeed an inefficient use of the available bandwidth, especially when the multimedia streams involved require a high packet sending rate. The optimization of these point-to-multipoint, or *multicast* communications thus requires one to send the same packet at most once on each link. Consequently, given a source and a set of destinations, the multicast routing minimizing the bandwidth consumption is not achieved on the shortest paths between the source and each destination but on a minimal set of links that connects the source and all destinations. This optimal set of links is called an *optimal Steiner Tree* in the Network Optimization terminology.

Unfortunately, support for multicast transmissions at the IP layer has not been completely standardized so far and has thus experienced a slow deployment within the Internet ([22]). Consequently, some applications that would suffer from bandwidth limitations by solely relying on IP routing have to implement their own multicast-like mechanisms. In this case, the nodes where the application is run can also be seen as the nodes of an *overlay network*, i.e a virtual network whose links are actually paths at the IP layer. Moreover by addressing the multicast issues at the application layer, the multicast architecture can be tailored to each particular application. For example, [2] tackles the optimization of a multicast overlay network that Akamai Technologies uses for video streaming applications and where redundant paths are added to augment the failure resilience of the architecture. Remark that, from the previous discussion, moving the

multicast mechanisms from the IP layer to the application layer can again be seen as an application of the end-to-end argument. Indeed it alleviates the tasks devolving on IP routers and allows applications to optimize the multicast functionality according to their own requirements ([12]).

In the first part of this thesis, we focus on applications which require a combination of unicast and multicast communications. This study is motivated by multimedia applications such as videoconferences in which user-dependent information has to be sent via unicast routing to a *core* node to be chosen, and then global information has to be multicast back from the core node to the users. Therefore the routing optimization now seeks a core location in the network so as to minimize a combination of the shortest paths from the users to the core and of a Steiner tree connecting the core and the users.

1.2.2 Automatic Repeat reQuest vs Forward Error Correction

As seen in Section 1.1, the prevalent mechanism used to protect applications from packet losses in the internet is the *Automatic Repeat reQuest (ARQ)* mechanism which consists in explicitly asking the sender to retransmit a packet when the receiver considers that this packet has been lost. Whereas this mechanism is well suited to provide users with a sufficient service for classical applications, real-time multimedia communications may be seriously impaired by retransmissions. The first reason is that they can potentially increase packet transmission delays by several round-trip times, resulting in an unacceptable communication latency. Secondly multimedia content such as video or audio is usually organized in a hierarchy of importance layers. Whereas packets from the first layers contain essential data, packets from the last layers usually contain detail information only. Thus losses of the latter packets can be tolerated to a certain extent. Therefore a key feature of multimedia applications is that they generally prefer to trade off the guaranteed transmission of less important packets against a delay decrease. Finally, ARQ mechanisms may be of limited interest in some environment such as multicast communications. In this particular case, a one-to-many reliable transmission can potentially cause a feedback implosion problem if too many receivers send notifications towards the sender. Moreover, because different receivers may experience different loss patterns and transmission delays, some receivers typically request a retransmission for packets that other have already received. This globally increases the average transmission delay experienced and can also lead to wasteful bandwidth usage.

Forward Error Correction (FEC) coding techniques protect data transmission from packet losses without the retransmission burden of ARQ by inserting redundant data inside the packet stream. *FEC block codes* with parameters (N, K) are characterized as follows : the sender's FEC encoder partitions the packet stream in blocks of K packets, called *data blocks* and adds $R = N - K$ redundant packets to each of these blocks. The so formed $N-$packets blocks depicted in figure 1.1 are called *FEC blocks*. The key property required for this coding technique is that, for each FEC block, the receiver's decoder is able to reconstruct the original data block from any K packets of the FEC block. Conversely, when more than R packets are lost, some data is irrecoverably lost.

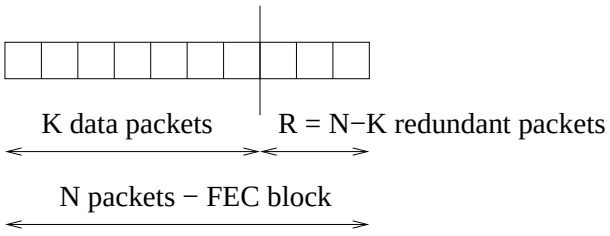

Figure 1.1: $(N, K)-$*FEC block*

Even though some losses within multimedia flows can thus be tolerated in principle, the situation is made worse by the fact that packet losses on the Internet are mostly due to congestion and thus arise in bursts instead of being uniformly distributed over time. It has been shown ([25]) that long sequences of consecutive packet losses significantly degrade the user-perceived quality of the received multimedia stream even if the average packet loss rate is low. Similarly FEC techniques lose their efficiency when the number of packets lost because of an error burst becomes larger than the number of redundant packets within a FEC block.

However it has been observed recently that splitting the transmission of FEC blocks on several channels usually decreases the impact of error bursts on the protection mechanism robustness ([46],[47],[48]). The intuition behind this fact is the following : if the traffic is split onto several paths, an error burst on one of the paths affects only a fraction of the number of packets it would have affected if all the traffic had been sent through that path. As discussed in Section 1.2.1 for multicast routing, the deployment of multipath routing techniques on the Internet is more likely to arise at the application layer than at the IP layer in the few next years. The second part of this thesis is devoted to the analysis of the effect of multipath

routing on the robustness of FEC techniques.

1.3 Thesis Contribution and Organization

As already mentioned, this thesis is subdivided into two independent parts. Part I concerns the problem of selecting a core node to be connected to a given set of users by unicast and multicast communications.

In Chapter 2, we address the problem of choosing this node so as to minimize the total bandwidth consumption of the communications. More precisely the problem is formulated as finding a core node which minimizes a weighted sum of two potentially competing criteria, one being the sum of the distances from the core node to a set of users, the other being the cost of connecting the core node and the terminals with a Steiner tree. We show that, in practice, we can rapidly compute a solution for which the objective is guaranteed to be within a few percents of the optimum. Chapter 3 is devoted to a rigorous analysis of the trade-offs between the two criteria. To state our result informally we show that there always exists a core location for which each criterion is close to its minimum value (if we were to disregard the other criterion). The results obtained are then used as an attempt to explain a posteriori why the algorithm described in Chapter 2 quickly finds a good core location.

In Part II, we first present in Chapter 4 the framework in which it has been shown that multipath routing can increase FEC robustness. Existing approaches characterize packet losses with Markovian models and seek a routing which minimizes the *probability of an irrecoverable loss (PIL)* for a given packet FEC block, that is the probability of loosing more than R packets within a block (see figure 1.1). We address the limitations of the results presented in the literature so far and address a method to extend them. In particular we argue that the FEC robustness is directly linked to the way packets losses are spread over time. Therefore we propose a method to approximate the probability of irrecoverable loss using a description of the packet flow called *peakedness*. Peakedness can be seen as measure of the distribution of packet traffic bursts over time. We present some computational results which show the effectiveness of our method by directly comparing the exact value of the PIL with our approximations for small network instances. In Chapter 5, we generalize the approach of Chapter 4 to a fluid traffic formulation of the problem. This alternative formulation allows us to propose a complete model and to address the FEC robustness problem for larger network instances. Note that a table

comparing the respective contributions of Chapters 4 and 5 is given at the end of Chapter 4.

Part I

The Core Location Problem

Chapter 2

Core Selection for Multimedia Applications

Many new multimedia applications on the Internet ask for the selection of a meeting point in the network. Such applications are for instance videoconferences or multi-player games. In both cases, each user sends his personal data to the meeting point. The role of this particular point is to gather the information received to create a single composite data flow which is multicast back to the users. More generally, there exists a class of multicast routing protocols, called *center-based* multicast protocols, which require an administrative center for each multicast group. In the *Core Based Tree* (CBT) protocol ([5]) for instance, if a source node wants to reach a multicast group, the data flow is first sent to a *core* node and then distributed to the group via a *shared tree*. Note that, in general, the source does not need to be a member of the multicast group. However we focus here on a particular case, called *All Receivers Sources* in [9], where the set of senders is (or can be approximated as) the set of receivers. As mentioned above, this case is relevant for several multimedia applications.

Empirical analysis has been carried out in order to measure the relationship between the choice of the core location and the performance of the routing scheme (see for instance [9], [23] and [58]). The performance of such a scheme is usually evaluated in terms of delay and bandwidth consumption. For the sake of simulation, the network is modeled as an undirected graph, the communication delay between two nodes of the network is approximated as the number of links on the path used and the bandwidth consumption is evaluated as the sum of the data flow on each link. More detailed models use weighted graphs, where each link

has a given weight which represents either its communication latency or its utilization cost (per bandwidth unit consumed).

If the objective is to minimize the average delay between every pair of users in the group, the optimal routing is achieved on a shortest path tree rooted at the core. In that case, each user-core shortest path is used in both directions and the optimal location of the core is such as to minimize the sum of the shortest path lengths between the core and every user. In Location Theory terminology, we seek the *1-median* of the users.

However, in order to minimize the bandwidth consumption, we must distinguish two different problems. The first one concerns the optimization of the unicast forward paths between the users and the core. Since each corresponding data flow is particular to its sender, we must optimize each user-core path. Therefore the total bandwidth consumption of these paths is minimized when the core is located at the 1-median of the users. The second problem is the optimization of the multicast flow from the core to all the users. Since the core has to send the same data to all the users, minimizing the bandwidth consumption is equivalent to minimizing the number of edges of the shared tree spanning the users and the core. In other words, we must find a *minimum Steiner tree* spanning the users and the core. For this problem, an optimal location for the core is any node on the optimum Steiner tree spanning the users only.

In this chapter we focus on the problem of choosing a core location in order to minimize the total bandwidth consumption of a CBT multicast communication, under the assumption that all receivers are also sources [9]. Consider, for example, a videoconferencing session in which every terminal sends a data flow to the core consuming say one unit of bandwidth, and the core then multicasts a common virtual scene of bandwidth λ back to the terminals. Typically, λ is an increasing function of the number of terminals and might not even be known in advance. In the *Minimum Bandwidth Cost Core Location Problem (MBCCL)*, the goal is to choose a core v so as to minimize the sum of the distances between v and the terminals plus λ times the Steiner tree cost connecting the core v and the terminals. This problem was first studied in [40]. The authors proposed an enumeration algorithm which essentially evaluates the objective function at every potential core location. Observe that simply evaluating this objective function for a given core location is already computationally hard since it requires the computation a minimum cost Steiner tree, which is known as an NP-hard problem ([33]). The goal of this chapter is to propose an algorithm which finds a good core location without exploring all

the nodes of the network. At the heart of the procedure, we use an efficient dual-ascent procedure ([63]) to approximate the Steiner trees. A nice feature of the chosen method is that, on termination of the algorithm, it provides us with a quality measure of the solution returned.

The remainder of this chapter is organized as follows. In Section 2.1, we give the definitions and formulate our main problem. Section 2.2 essentially contains an overview of the algorithm that we use to approximately compute minimum cost Steiner trees. Next we present in Section 2.3 our algorithm to select a core location that approximately solves the MBCCL problem. Computational results, given in Section 2.4, show that our procedure is rapid and gives good performance guarantees. In other words, it allows us to efficiently compute a good core location by exploring only a small fraction of the network nodes, as desired.

2.1 Definitions and Problem Formulation

Let $G = (V, E, c)$ be a connected, undirected graph with edge costs $c : E \rightarrow \mathbb{R}^+$ and $T \subseteq V$ be a particular non-empty subset of vertices called *terminals*. G naturally models the communication network : the vertex set V represents network nodes and the edge set E represents communication links between those nodes. For each link e, the edge cost c_e may be seen as the resource consumption cost of a flow of unitary bandwidth going across link e. Finally, the terminal set T corresponds to the users involved in the multimedia application considered. Although we have considered unit costs in the introduction, we carry out our analysis with arbitrary positive edge costs.

The *cost* of a subgraph H of G is the sum of the costs of all edges in H:

$$c(H) = \sum_{e \in E(H)} c_e.$$

The shortest path distance according to the edge costs c_e defines a metric on the vertex set V. The distance between vertex i, j is denoted by $d(i, j)$. For a subgraph H of G, $d_H(i, j)$ denotes the distance between i and j in H.

For any $v \in V$ and $S \subseteq V$, we denote the sum of the distances between v and the vertices in S by

$$f_S(v) = \sum_{t \in S} d(v, t).$$

A *1-median* for S is a vertex minimizing f_S. Whenever $S = T$, we simply write $f(v)$ for $f_T(v)$. Let v^* denote a 1-median for the terminal set T.

For any $S \subseteq V$, a *Steiner tree* for S is a tree spanning S. The cost of a minimum cost Steiner tree for S is denoted by $ST(S)$.

As announced in the introduction of this chapter, given a bandwidth parameter λ, the total cost for a chosen core location v is given by the function

$$cost(v) = f(v) + \lambda ST(T \cup \{v\}).$$

The *Minimum Bandwidth Cost Core Location (MBCCL)* problem is then defined as the problem of finding a vertex v^{OPT} for which

$$cost(v^{OPT}) = \min_{v \in V} cost(v).$$

2.2 Computation of the Cost Function

In comparison to the Steiner tree problem, the function $f(v)$ is easily computed in polynomial time using one of the classical shortest path algorithms ([6],[17]). Moreover these shortest path costs are in practice available in routing tables. Therefore in this chapter and the next, we suppose that these costs are available and do not discuss their computation further.

As already mentioned, unlike the shortest path problem, computing a minimum cost Steiner tree is a NP-hard problem ([33]). Consequently many heuristics and approximation algorithms have been devised in the literature. Up to now the best known approximation algorithm is due to [50] and achieves a performance guarantee approaching $1 + \ln \frac{3}{2} \approx 1.55$, this value being an upper bound on the ratio of the computed tree cost and the optimal cost. A survey of existing algorithms up to 1992 can be found in [32]. An extensive comparison of most of these algorithms was performed in [60]. There the algorithm proposed by Wong in [63] is reported to achieve the best solution in most cases, although no theoretical performance guarantee is known for this procedure.

The remainder of this section is devoted to an overview of Wong's method. Instead of giving a complete description of the algorithm, we focus here on its main characteristics. First of all, let us point out that the algorithm was originally devised for directed graphs. However it can immediately be used for our undirected graph G if we consider that each edge of G stands for two arcs in opposite directions. More precisely , we

turn $G = (V, E, c)$ into its directed version, $G_d = (V, A, c)$, which is obtained by replacing each edge $e = \{i, j\}$ in G by two arcs (i, j) and (j, i). Hence $A = \{(i, j) \in V \times V \mid \{i, j\} \in E\}$. Edge costs naturally induce arc costs as follows : for all $(i, j) \in A$, $c_{ij} = c_{\{i,j\}}$. Moreover we choose an arbitrary terminal vertex $r \in T$ called the *root*. We now seek a minimum cost acyclic subgraph of G_d in which there is a directed path from r to any vertex in $T \setminus \{r\}$.

Wong's algorithm is based on the following mixed integer program formulation of the latter problem.

$$(P) \quad \min z_P = \sum_{(i,j)\in A} c_{ij}\, y_{ij}, \tag{2.1}$$

subject to (2.2)

$$\sum_{j\in\delta^-(i)} x^t_{ji} - \sum_{j\in\delta^+(i)} x^t_{ij} = \left\{ \begin{array}{ll} -1, & \text{if } i = r, \\ 1, & \text{if } i = t, \\ 0, & \forall i \in V \setminus \{r, t\}, \end{array} \right\} \forall\, t \in T \setminus \{r\} \tag{2.3}$$

$$x^t_{ij} \leq y_{ij}, \quad \forall\, (i, j) \in A,\ t \in T \setminus \{r\}, \tag{2.4}$$

$$x^t_{ij} \geq 0, \quad \forall\, (i, j) \in A,\ t \in T \setminus \{r\}, \tag{2.5}$$

$$y_{ij} \in \{0, 1\}, \quad \forall\, (i, j) \in A, \tag{2.6}$$

where, for any vertex $v \in V$, $\delta^+(i) = \{a = (i, j) \in A | j \in V\}$ denotes the set of arcs coming from vertex i and $\delta^-(i) = \{a = (j, i) \in A | j \in V\}$ denotes the set of arcs going to vertex i. For each arc $a \in A$, y_a indicates whether we select a as part of the chosen Steiner directed tree. To ensure that the selected arcs connect r with each other terminal $t \in T \setminus \{r\}$, we add variables x^t_{ij}, which represent the value on (i, j) of a flow from r to t, and equations (2.3)-(2.5), which guarantee that a unit of flow can be sent from r to t using only arcs a with $y_a = 1$ for all $t \in T \setminus \{r\}$.

Now consider (LP), the linear relaxation of (P), obtained by replacing the integrality constraints (2.6) by the relaxed bound constraints $0 \leq y_{ij} \leq 1$ for all $(i, j) \in A$. Since it has a number of variables and constraints polynomial in the input size, (LP) is solvable in polynomial time. Unfortunately (LP) does not have an optimal solution with integral values for all y_{ij} in general. Hence its optimal objective value, z^{OPT}_{LP}, is generally strictly less than the optimal objective value of the initial mixed integer program (P), z^{OPT}_P. However practical experience reveals that z^{OPT}_{LP} is typically very close to z^{OPT}_{LP}. The worst known examples have a ratio $\frac{z^{OPT}_P}{z^{OPT}_{LP}}$ which can be made arbitrarily close to $\frac{8}{7}$ ([30]).

Even though (LP) can in principle be exactly solved in polynomial time, the number of variables and constraints can be up to $O(|V|^3)$ for a dense graph and a number of terminals proportional to $|V|$. Therefore computing an exact solution of (LP) might already be computationally expensive. For that reason, Wong proposed to first compute a "good" feasible solution of (DLP), the linear programming dual of (LP), which may be written as follows.

$$(DLP) \quad \max \sum_{t \in T \setminus \{r\}} (v_t^t - v_r^t), \tag{2.7}$$

$$\text{subject to} \quad v_j^t - v_i^t \leq w_{ij}^t, \quad \forall\, t \in T \setminus \{r\},\ (i,j) \in A \tag{2.8}$$

$$\sum_{t \in T \setminus \{r\}} w_{ij}^t \leq c_{ij}, \quad \forall\, (i,j) \in A, \tag{2.9}$$

$$w_{ij}^t \geq 0 \quad \forall\, t \in T \setminus \{r\},\ (i,j) \in A. \tag{2.10}$$

Dual variables v_i^t correspond to the flow conservation equations (2.3) at vertex i and dual variables w_{ij}^t correspond to equations (2.4). By the linear programming strong duality theorem, the optimal objective of (DLP), z_{DLP}^{OPT} equals z_{LP}^{OPT}.

Wong's algorithm consists of the following two main steps :

1. **dual-ascent procedure :** the algorithm starts with the trivially feasible solution where all dual variables set to 0. Then it greedily tries to increase the dual objective up to a value $z_{DLP}^{feasible}$ hopefully close to the optimal value z^{DLP}.

2. **heuristic for the Steiner tree** The algorithm constructs a subgraph H of G_d based on the dual-solution obtained. H is guaranteed to connect r to all other terminals. Then it computes a minimum cost directed spanning tree in H rooted at vertex r. Finally edges which do not disconnect r from other terminals are removed from this spanning tree and the algorithm outputs the resulting tree. As the returned tree can interpreted as a feasible solution of the initial mixed integer program, we denote its cost by $z_P^{feasible}$.

A key feature of Wong's algorithm is that it provides us not only with a feasible Steiner tree of cost $z_P^{feasible}$ but also with a lower bound $z_{DLP}^{feasible}$ on the optimal cost z_P^{OPT}. Therefore, upon completion of the algorithm,

we are able to give a measure of the quality of the Steiner tree that is generated. As depicted in figure 2.1, the total gap between the upper bound $z_P^{feasible}$ and lower bound $z_{DLP}^{feasible}$ can be decomposed in the following three parts :

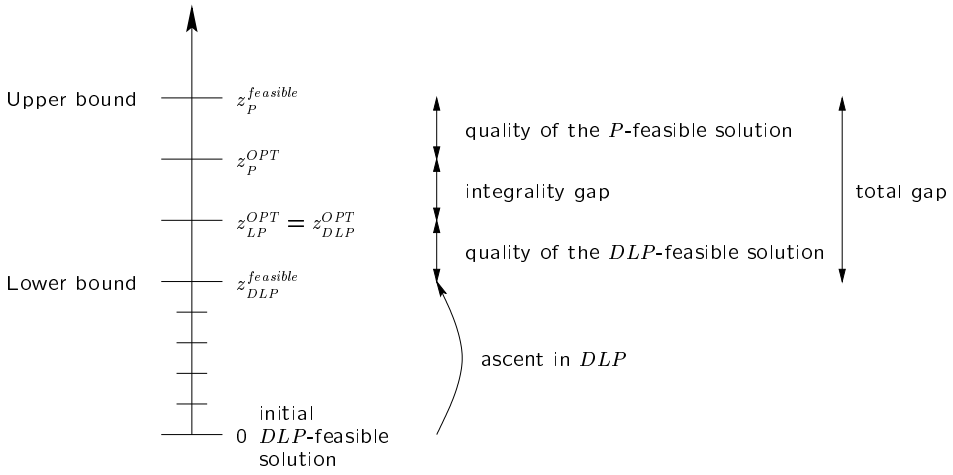

Figure 2.1: *Decomposition of the total gap* $z_P^{feasible} - z_{DLP}^{feasible}$ *for Wong's algorithm.*

1. the gap $z_P^{feasible} - z_P^{OPT}$ which is the true, but usually unknown, measure of quality of the solution obtained.

2. the integrality gap $z_P^{OPT} - z_{LP}^{OPT}$ of the chosen mixed integer program, which is somehow inherent to the computational intractability of the Steiner tree problem,

3. the gap $z_{DLP}^{OPT} - z_{DLP}^{feasible}$ which measures how far away one is from an optimal solution of (DLP) when the dual-ascent procedure stops.

Because of two last gaps, the quality measure for the solution computed is generally overpessimistic. However it is difficult to know a priori how the total gap is shared among these three components.

Finally observe that, in order to solve an instance of the MBCCL problem, one has to compute Steiner trees spanning the terminal set T and a potential core location v. Since different core locations v are naturally considered in the course of the algorithm described in next section, it is worth noting that a feasible solution of (DLP) for T may be trivially extended to

a feasible solution of (DLP) for $T \cup \{v\}$ by setting extra variables to zero. Therefore we speed up the computation of a Steiner tree for $T \cup \{v\}$ by Wong's algorithm by initializing the dual-ascent procedure with the final values obtained for T.

2.3 Algorithm for the Minimum Bandwidth Cost Core Location Problem

We present below an algorithm seeking an optimal core location, that is a vertex v^{OPT} satisfying $cost(v^{OPT}) = \min_{v \in V} cost(v)$. Since we use Wong's algorithm to compute a Steiner tree, our algorithm has been designed so as to also provide us with a lower bound on the optimal objective value of the MBCCL problem.

2.3.1 Algorithm Description

Let us start by briefly presenting the main ideas of the algorithm. In the first step, we attempt to find an optimal Steiner tree spanning $\mathcal{ST}_W$ using Wong's procedure. Then we select a vertex of this Steiner tree, denoted by v_s in the description below, which minimizes $f(v)$. Finally we explore vertices not in $\mathcal{ST}_{\mathcal{W}}$ that, if chosen as the core, could yield a lower total cost.

Since we do not compute the exact cost of an optimal Steiner tree, we can only obtain bounds on the value of $cost(v)$ for any vertex v. Therefore we define the following upper and lower bounds :

$$\begin{aligned} \overline{cost}(v) &= f(v) + \lambda \overline{ST}(T \cup \{v\}), \\ \underline{cost}(v) &= f(v) + \lambda \underline{ST}(T \cup \{v\}). \end{aligned}$$

During execution of the algorithm, we maintain $\overline{v}$ as the vertex with the lowest value of $\overline{cost}(\overline{v})$ among the vertices already explored. Let $\overline{cost} = \overline{cost}(\overline{v})$. Hence $\overline{cost}$ is clearly an upper bound on $cost^{OPT}$. We also maintain the variable $\underline{cost}$ as a lower bound on $cost^{OPT}$.

We can now describe the algorithm that we call Steiner Tree First.

STF
input: A graph $G = (V, E, c)$ and a set of terminals $T \subseteq V$.
output: A core location, a lower and an upper bound on $cost^{OPT}$.

1. Order $V = \{v_1, v_2, \ldots, v_{|V|}\}$ so that $f(v_1) \leq f(v_2) \leq \ldots \leq f(v_{|N|})$.

2. Compute a Steiner tree $\mathcal{ST}_W$ spanning T with Wong's algorithm giving bounds on the optimal cost : $\overline{ST}(T)$ and $\underline{ST}(T)$.

   Let $v_s := \arg\min_{v \in \mathcal{ST}_W} f(v)$.

   Initialize :

$$\begin{aligned} \overline{cost} &:= \overline{cost}(v_s) = f(v_s) + \lambda \overline{ST}(T), \\ \overline{v} &:= v_s, \\ \underline{cost} &:= f(v_1) + \lambda \underline{ST}(T), \\ t &:= 1. \end{aligned}$$

3. Compute $\overline{ST}(T \cup \{v_t\})$ and $\underline{ST}(T \cup \{v_t\})$.

   Update :

$$\begin{aligned} \overline{cost} &:= \min\{\overline{cost}(v_1), \ldots, \overline{cost}(v_t), \overline{cost}(v_s)\}, \\ \overline{v} &:= \arg \min_{v \in \{v_1, \ldots, v_t\} \cup \{v_s\}} \overline{cost}(v), \\ \underline{cost} &:= \min\{\underline{cost}(v_1), \ldots, \underline{cost}(v_t), f(v_{t+1}) + \lambda \underline{ST}\}, \end{aligned}$$

   where the last term is thus a lower bound on $cost(v)$ for all vertices $v \in V \setminus \{v_1, \ldots, v_t\}$.

4. **If** $f(v_{t+1}) + \lambda \underline{ST}(T) \geq \overline{cost}$ **or** $t = s - 1$
   then STOP and **return** $\overline{v}$, $\underline{cost}$ and $\overline{cost}$.
   else $t = t + 1$ and go back to step 3.

We now justify our first stopping criterion in step 4 : $f(v_{t+1}) + \lambda \underline{ST}(T) \geq \overline{cost}$. If the inequality is satisfied, the lower bound on $cost(v)$ for vertices v not yet treated is higher than the actual best feasible solution. Therefore none of these vertices can yield a better feasible solution.

2.3.2 Properties

We first give two easy properties of *STF*.

Proposition 2.1. *On termination of* STF*,*

(i) $\overline{cost} = \min_{i=1,\ldots,s} \overline{cost}(v_i)$,

(ii) $\underline{cost} = \min_{i=1,\ldots,s} \underline{cost}(v_i)$.

Proof. (i) is true by construction.

(ii) On termination of *STF*, if $t = s - 1$,

$$\begin{aligned} \underline{cost} &= \min\{\underline{cost}(v_1), \ldots, \underline{cost}(v_{s-1}), f(v_s) + \lambda \underline{ST}(T)\} \\ &= \min_{i=1,\ldots,s} \underline{cost}(v_i). \end{aligned}$$

Otherwise

$$\begin{aligned} f(v_{t+1}) + \lambda \underline{ST}(T) &\geq \overline{cost} \\ &= \overline{cost}(\overline{v}) \\ &\geq \underline{cost}(\overline{v}). \end{aligned}$$

Thus, for all $i > t$,

$$\underline{cost}(v_i) \geq f(v_{t+1}) + \lambda \underline{ST}(T) \geq \underline{cost}(\overline{v}).$$

Therefore $\underline{cost} = \min_{i=1,\ldots,s} \underline{cost}(v_i)$.

□

The following results gives us the link between a quality guarantee for the Steiner tree costs and a guarantee for the total cost.

Proposition 2.2. *On termination of* STF*,*

(i) if an exact algorithm is used to solve the Steiner tree problem, $\overline{v}$ is an optimal core location with respect to the MBCCL problem,

(ii) if the Steiner tree problems are solved with an approximation algorithm satisfying

$$\overline{ST}(Q) \leq (1 + \delta)\underline{ST}(Q)$$

for all $Q \subseteq V$,

$$\overline{cost} \leq \left(1 + \frac{\lambda\delta}{1 + \lambda}\right) cost.$$

Proof. (i) is trivial from Proposition 2.1.

(ii) Let $\underline{v} = \arg\min_{i=1,\dots,s} \underline{cost}(v_i)$. Then

$$\begin{aligned}
\underline{cost} &= \underline{cost}(\underline{v}) = f(\underline{v}) + \lambda \underline{ST}(T \cup \{\underline{v}\}), \\
\overline{cost} &\leq \overline{cost}(\underline{v}) = f(\underline{v}) + \lambda \overline{ST}(T \cup \{\underline{v}\}), \\
\frac{\overline{cost}}{\underline{cost}} &\leq \frac{f(\underline{v}) + \lambda \overline{ST}(T \cup \{\underline{v}\})}{f(\underline{v}) + \lambda \underline{ST}(T \cup \{\underline{v}\})} \\
&\leq \frac{f(\underline{v}) + \lambda(1+\delta)\underline{ST}(T \cup \{\underline{v}\})}{f(\underline{v}) + \lambda \underline{ST}(T \cup \{\underline{v}\})} \\
&= 1 + \frac{\lambda\delta \underline{ST}(T \cup \{\underline{v}\})}{f(\underline{v}) + \lambda \underline{ST}(T \cup \{\underline{v}\})} \\
&\leq 1 + \frac{\lambda\delta}{1+\lambda},
\end{aligned}$$

as $f(\underline{v}) \geq \underline{ST}(T \cup \{\underline{v}\})$. □

If our algorithm for the Steiner tree problem satisfies

$$\overline{ST}(Q \cup \{v\}) \geq \overline{ST}(Q) \tag{2.11}$$

for all $Q \subseteq V$ and $v \in V$, then $\overline{cost}(v_t) \geq \overline{cost}(v_s)$ for all $t \geq s$. Thus it is useless to consider vertex v_t with $t \geq s$. This motivates the second stopping criterion.

The use of Wong's algorithm to compute a Steiner tree leads us to make the following remarks.

1. Though Wong's algorithm does not have a known performance guarantee, numerical experiments ([32], [60], [63] have shown that it usually terminates with a very small relative gap between the lower and the upper bounds on the optimal Steiner tree cost. Hence, according to Proposition 2.2, using Wong's algorithm within *STF* should provide us with good solutions for the MBCCL problem as well.

2. Wong's algorithm does not satisfy the property given by equation (2.11) : if we constrain the Steiner tree to span an extra vertex v, Wong's algorithm might actually end up with a tree of cost $\overline{ST}_W(Q \cup \{v\}) < \overline{ST}_W(Q)$. Note that, in such a case, the output of Wong's algorithm for $Q \cup \{v\}$ is naturally also a new and better approximate solution of the optimal Steiner tree spanning Q only. Consequently, the second stopping criterion $t \geq s$ may in principle prevent us to consider a potentially better core location v_t with

$t \geq s$. In the next section, we will indeed observe that, with Wong's algorithm, removing that stopping criterion yields slightly better solutions but at the price of a much larger number of explored vertices.

2.4 Computational Results

Here we first describe our graph generation method and then we present the results of our experiments using the *STF* algorithm, as well as a variant in which the second stopping criterion is removed.

2.4.1 Problem Instances Generation

We have used the LEDA C++ library to generate random graphs. The details of the routine used are given in [45]. Here are the parameters used in generating the instances :

1. $|V|$, the number of vertices, was fixed at 100 in all instances.
2. $|E|$, the number of edges, varied from its minimum (99) to its maximum value (4950).
3. The edge costs were integers chosen randomly between 1 and 100.
4. The number of terminals $|T|$ ranged from 2 to $|V| = 100$. Note that, for the extreme values of T, the *STF* algorithm always finds an optimal solution since the Steiner tree problem then reduces to the shortest path problem and the minimum spanning tree problem respectively.
5. We chose the values 1, 3 and 5 for the bandwidth λ used in the virtual scene broadcast. For larger values of λ, the cost of the Steiner tree dominates the cost of the shortest path tree.

2.4.2 Performance Measure

The performance is measured in two ways :

1. The *quality guarantee ratio* of the solution defined as

$$q = \frac{\overline{cost} - \underline{cost}}{\underline{cost}}.$$

Below this ratio is expressed as a percentage.

2. The number of potential core vertices explored in step 3 of *STF*.

For each combination of values for the parameters described above, 100 instances were generated in order to obtain an average value of the performance measures. The average quality guarantee ratio is denoted by $\bar{q}$. Each problem instance was solved by the *STF* algorithm and its modified version without an imposed termination before vertex v_s. The latter is denoted by *STF'*. In the following observations, we focus on instances with $\lambda = 5$. Similar results were obtained with the values $\lambda = 1$ and 3.

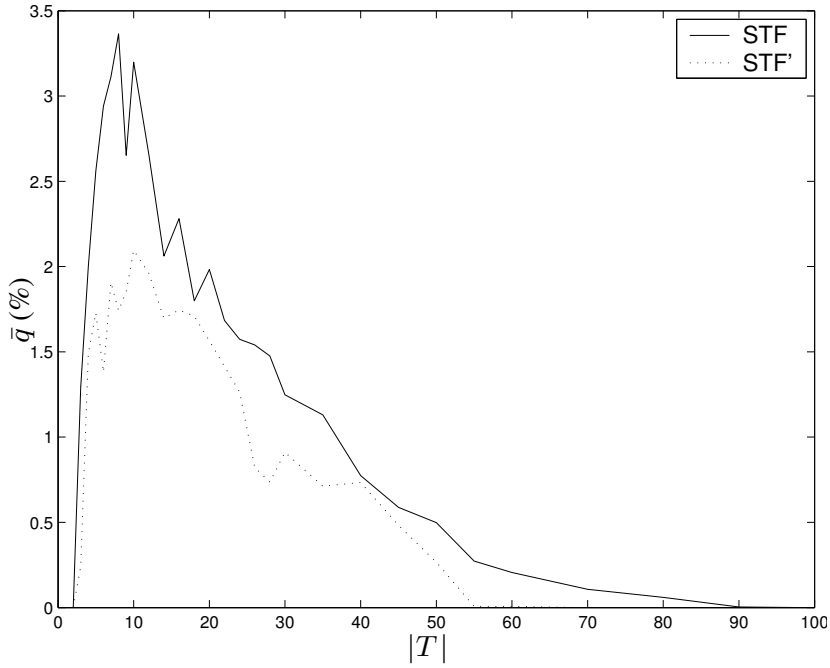

Figure 2.2: *Average quality guarantee $\bar{q}$ vs number of terminals $|T|$ for $|V| = 100$, $|E| = 4950$ and $\lambda = 5$*

Figure 2.2 shows a plot of the average quality guarantee ratio $\bar{q}$ for *STF* (continuous curve) and *STF'* (dotted curve) against the number of terminals $|T|$. $|E|$ is set to its worst case value : $|E| = 4950$. For both algorithms, we observe that the worst-case values for $|T|$ range between 5 and 20 terminals approximately. *STF'* performs slightly better than *STF*, showing that there are some gains from going further than vertex v_s. Figure 2.3 shows a plot of q for *STF* and *STF'* against the number of edges $|E|$. Note that the average is here computed for $|T|$ ranging from 2 to 20. We

can see that the main increase of this ratio occurs up a density of about 200 edges. Then the increase is roughly linear until one obtains complete graphs. Once again we observe the better performance of *STF'* for every value of $|E|$.

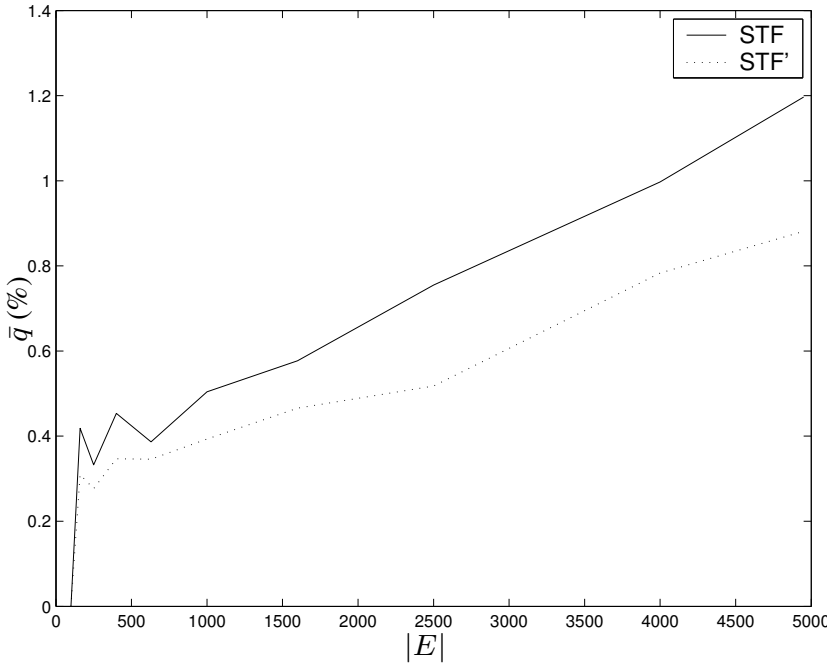

Figure 2.3: *Average quality guarantee* $\bar{q}$ *vs number of edges* $|E|$ *for* $|V| = 100$, $2 \leq |T| \leq 20$ *and* $\lambda = 5$

However *STF'* explores a much larger number of vertices than *STF*. Figures 2.4 and 2.5 show the average number of potential core locations explored by *STF* and *STF'* respectively for the same problem instances as in figure 2.2 (E=4950, $\lambda = 5$). One can observe that the average number of vertices explored by *STF'* can increase up to 9.5, while this average is always under 0.13 for *STF*. By looking separately at all instances generated, it turns out that the maximum number of vertices explored by *STF* is 5 whereas this maximum is 48 for *STF'*. These observations may interpreted as follows : on the one hand, *STF* needs to compute only one Steiner tree for most instances and, on the other hand, for the most "difficult" instances (i.e. when $|T| \approx 10$), the gap between the bounds computed by Wong's algorithm may force *STF'* to explore many potential core locations

outside the initial Steiner tree.

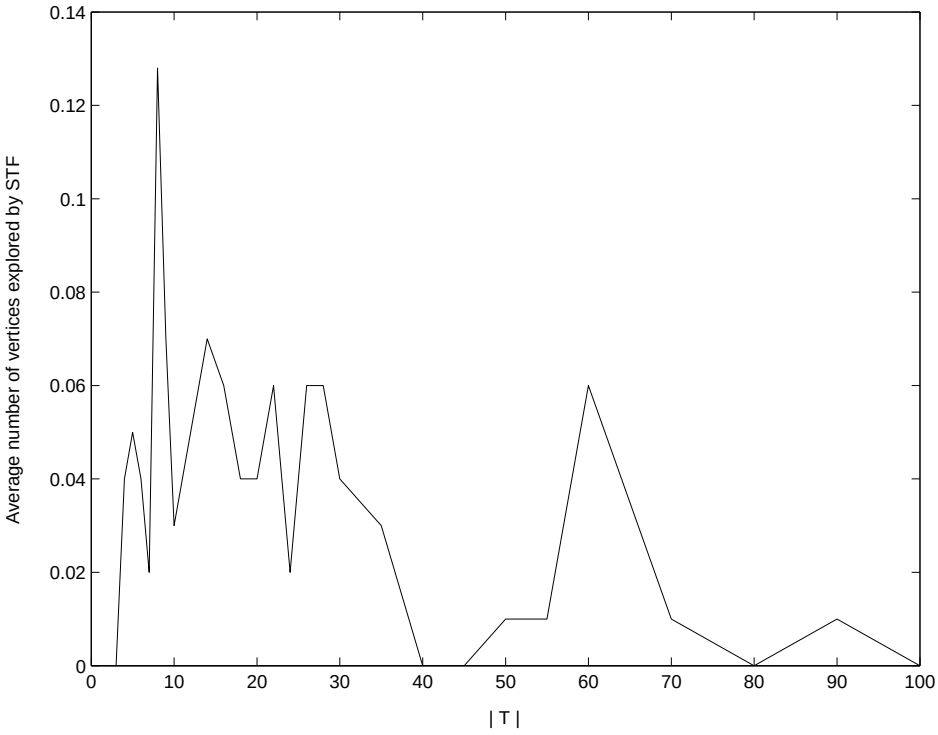

Figure 2.4: *Average number of vertices explored by* STF *vs number of terminals* $|T|$ *for* $|V| = 100$, $|E| = 4950$ *and* $\lambda = 5$

Another quantity of interest is the fraction of the total running time spent by the algorithm to explore vertices outside the initial Steiner tree (and to evaluate the $cost$ function). This quantity actually behaves very similarly to the number of explored vertices as in figures 2.4 and 2.5. For the same values of the parameters as above, the average fraction for *STF* is at most 0.035 whereas it is at most 0.12 for *STF'*. The execution of step 3 takes thus a significant part of the total running time for *STF'*.

Naturally, the numerical results presented here depends on the method used to generate random instances. We have also run experiments with generators of graphs having internet-like topologies (see [19] for instance). In all cases, the performances of our algorithms were even better than the ones mentioned above. We believe that this is partly explained by the fact that these graphs typically have a low connectivity. However it is not clear whether the performances detailed in this section may be con-

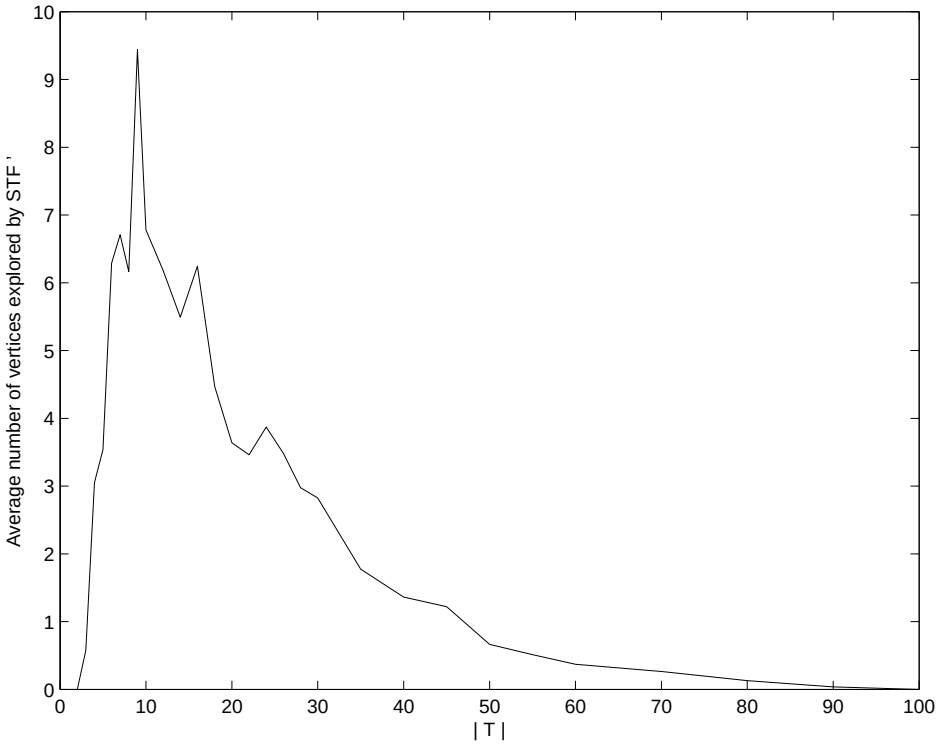

Figure 2.5: *Average number of vertices explored by* STF′ *vs number of terminals* $|T|$ *for* $|V| = 100$, $|E| = 4950$ *and* $\lambda = 5$

sidered as the worst cases. For instance, our choice of complete graphs and edge costs implies that the shortest paths between arbitrary vertices consist of very few edges most of the time. Hence optimum Steiner trees have a number of Steiner vertices (i.e. vertices which are not terminals) that grows almost linearly with $|T|$ for small values of $|T|$. This is depicted in figure 2.6 with a plot of the average number of Steiner vertices in the tree computed by STF against $|T|$.

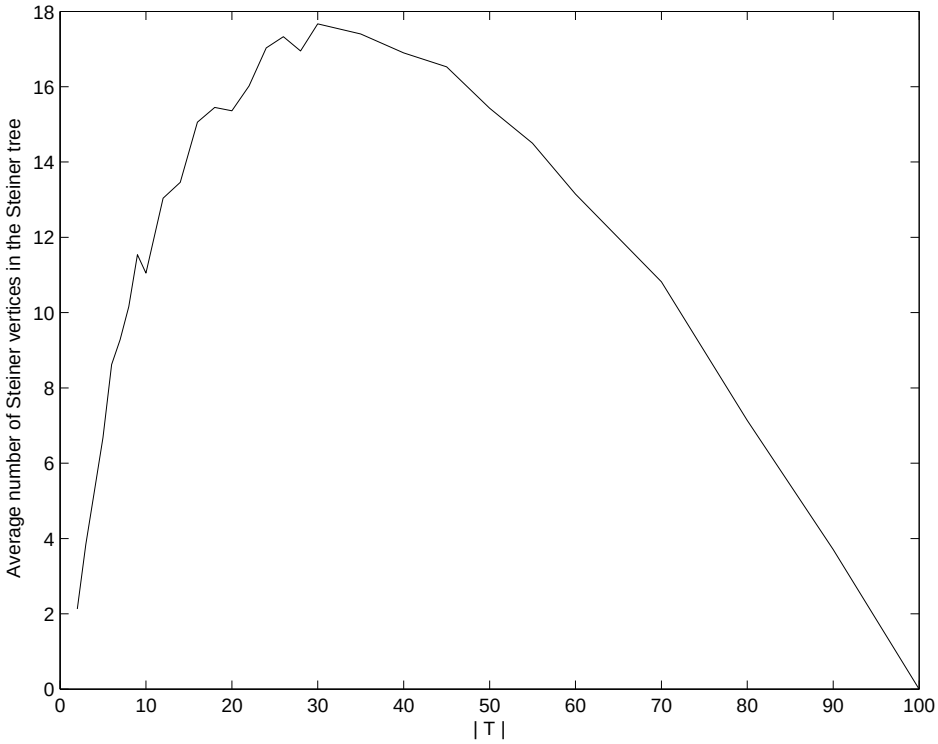

Figure 2.6: *Average number of Steiner vertices in the Steiner tree computed by* STF *vs number of terminals* $|T|$ *for* $|V| = 100$, $|E| = 4950$ *and* $\lambda = 5$

2.5 Conclusion

We have described in this chapter an algorithm to approximately solve the Minimum Bandwidth Cost Core Location problem. We have also derived a performance guarantee assuming that the procedure for the computation of Steiner tree has a known performance guarantee. In our experiments,

the solution always has a quality guarantee below a few percent, which is actually not surprising since we have used Wong's algorithm which is known to solve the Steiner tree problem very close to optimality in practice.

More interestingly, we have observed that the algorithm can rapidly obtain good solution without exploring many potential core locations. In fact, for most instances, only one Steiner tree is computed. This can be interpreted as saying that the algorithm quickly finds a core location such that the 1-median and the Steiner criteria are simultaneously close to their minimum values.

In the next chapter, we compute absolute bounds on the trade-offs between these two criteria. We then apply our result to show that it is indeed always possible to compute solutions of the MBCCL with an a priori approximation guarantee by computing a single Steiner tree.

Chapter 3

Trade-offs Between the 1-median and Steiner Criteria

In the previous chapter, we have considered the problem of locating a core node v under two criteria, one being the sum of the distances between v and the terminals (the "median" criterion), the other being the cost of the Steiner tree linking v to the terminals (the "Steiner tree" criterion). As we have seen, the 1-median might not be located on an optimum Steiner tree connecting the terminals and therefore there is some trade-off between the two criteria. Instead of considering a weighted sum of the two criteria as in Chapter 2, we present here a bicriteria analysis which consider each criterion as an individual objective. The main contribution of this chapter is to give bounds on the trade-off between these two competing objectives. To state our result informally, we show that either the optimum 1-median can be connected to one of the terminals without increasing much the cost of the Steiner tree connecting the terminals, or there exists a terminal for which the median criterion is not much more than for the optimum 1-median. More precisely, let α be the ratio for the median criterion between the best terminal and the optimum 1-median, and let β be the ratio of the Steiner tree cost for connecting the terminals and the optimum 1-median to the Steiner tree cost for connecting just the terminals. We can easily show that $\alpha \leq 2$, $\beta \leq \frac{3}{2}$, and that these values can asymptotically be attained. More interestingly, we show that for any $(\bar{\alpha}, \bar{\beta})$ ($1 \leq \bar{\alpha} \leq 2$ and $1 \leq \bar{\beta} \leq \frac{3}{2}$) such that $\bar{\alpha} = 3 - \frac{1}{2-\bar{\beta}}$ (the *trade-off function*, see Figure 3.1), either $\alpha \leq \bar{\alpha}$ or $\beta \leq \bar{\beta}$. Restating this, if connecting the optimum 1-median to a Steiner tree on the terminals increases its cost by a factor greater than $\bar{\beta}$ then we can select one of the terminals as the core and the median criterion

is no more than $\bar{\alpha}$ times the optimum 1-median value. We actually give a slightly better trade-off function between α and β in Theorem 3.1; this is indicated by a dotted line on Figure 3.1. Theorem 3.1 for example implies that either α or β is at most $(1+\sqrt{3})/2 < 1.37$. We would like to mention that even simply showing that β is close to 1 whenever α is close to 2 does not appear to be trivial.

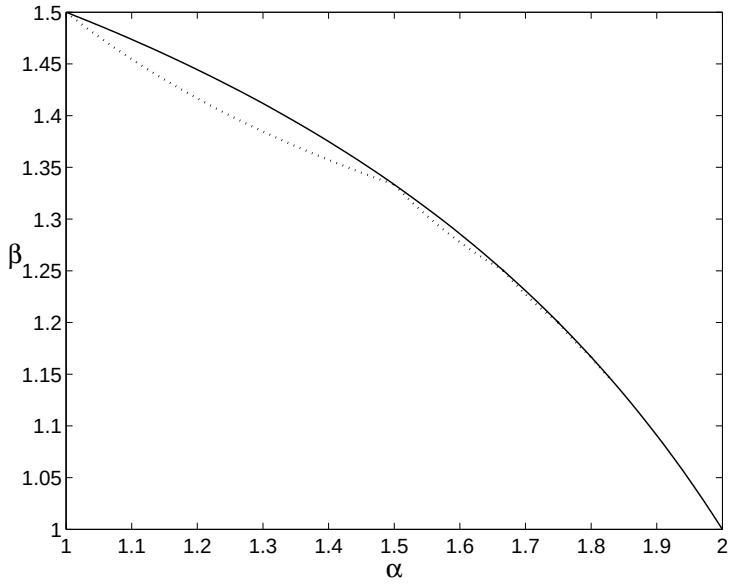

Figure 3.1: *Trade-off functions*

In terms of bicriteria guarantees, our result can be interpreted as saying that, for any $(\bar{\alpha}, \bar{\beta})$ on the trade-off function, there exists a core node such that its median criterion is within $\bar{\alpha}$ of the optimum 1-median and the cost of connecting it to the terminals is at most $\bar{\beta}$ times the cost of connecting only the terminals. In fact, the above discussion says that there exists a core node with either a $(1, \bar{\beta})$ guarantee (if we end up selecting the 1-median), or with a $(\bar{\alpha}, 1)$ guarantee (if we end up selecting one of the terminals). Bicriteria results have been given for many combinatorial optimization problems, see e.g. [38], [43] and [54]. In particular, some problems dealing with trade-offs between shortest paths and spanning tree costs have been studied in [4] and [39].

Finally our results can be applied to the MBCCL problem introduced in the previous chapter. Note that we do not require here the bandwidth parameter λ to be known in advance. If we use a ρ-approximation algorithm

to compute a Steiner tree, we clearly obtain a ρ-approximation algorithm for the MBCCL problem if, as in [40], we enumerate all possible core locations. Note that a Steiner tree must then be computed for each location considered. However, we use our trade-off results to show that we can approximate the core location problem with a ratio of $1 + \frac{\rho^2}{4}$ (slightly worse than ρ) by approximating only one instance of the minimum cost Steiner tree problem. In particular, for ratio $\rho \approx 1.55$ obtained in [50], this gives 1.6, while for $\rho = 2$ corresponding to a minimum spanning tree heuristic for which there exist distributed algorithms, we also obtain a bound of 2.

The remainder of the chapter is organized as follows. Section 3.1 gives the basic definitions. Our trade-off function is derived in Section 3.2. After deriving basic inequalities in Section 3.2.1, we first prove in Section 3.2.2 the existence of a decomposition of any tree into a small number of subtrees, each of cost bounded by a fraction of the original cost of the tree; this is stated precisely in Proposition 3.1. The next step is to bound the sum of the distances from any terminal t to the other terminals by going through the optimum Steiner tree for those terminals within the same member of the decomposition as t is and going through the optimum 1-median for the other terminals; this is the essence of Proposition 3.2. We then apply our trade-off results to the Minimum Bandwidth Cost Core Location Problem and prove that approximating only one Steiner tree is sufficient to obtain a good approximation ratio. These results are presented in Section 3.3.

3.1 Definition of α and β Parameters

Note that we still use the network model and notation introduced in Section 2.1 throughout this chapter. In particular recall that v^* denotes a 1-median of the terminal set T, $f_S(t) = \sum_{t \in S} d(v,t)$, $f(t) = f_T(t)$ and $ST(S)$ is the cost of a minimum cost Steiner tree spanning S, for any $S \subseteq V$. Moreover we give here the exact definitions of the two criteria mentioned above.

We define $\alpha = \frac{f(v_T)}{f(v^*)}$, where $v_T = \arg\min_{v \in T} f(v)$. In the introduction, β was defined to be $\frac{ST(T \cup \{v^*\})}{ST(T)}$. In order to make this quantity more tractable, we observe that $ST(T \cup \{v^*\}) \leq ST(T) + d(v^*, T)$ where $d(v^*, T) = \min_{v \in T} d(v^*, v)$, and we actually define β to be $1 + \frac{d(v^*,T)}{ST(T)}$. Thus β is an upper bound on $\frac{ST(T \cup \{v^*\})}{ST(T)}$.

3.2 Relationship between α and β

Here we derive our main inequalities between the parameters α and β.

3.2.1 Basic Inequalities

We first start by deriving a series of very simple inequalities that in particular show that $\alpha \leq 2$ and $\beta \leq \frac{3}{2}$.

Lemma 3.1. *If $\mathcal{T}$ is a tree spanning $S \subseteq V$ then $\sum_{t \in S} f_S(t) \leq \frac{|S|^2}{2} c(\mathcal{T})$.*

Proof. We have

$$\sum_{t \in S} f_S(t) = \sum_{t,t' \in S} d(t,t') \leq \sum_{t,t' \in S} d_{\mathcal{T}}(t,t'),$$

where we upper bound each $d(t,t')$ by constraining the distance to be computed on $\mathcal{T}$. Observe that $d_{\mathcal{T}}(t,t')$ is the cost of the unique path in $\mathcal{T}$ between t and t'. If we consider these paths for all pairs $(t,t') \in S \times S$, each edge e of $\mathcal{T}$ is used $2\, n_e(|S| - n_e)$ times, where n_e is the number of vertices of S in a connected component of $\mathcal{T} \setminus e$. Hence

$$\sum_{t \in S} f_S(t) \leq 2 \sum_{e \in E(\mathcal{T})} n_e(|S| - n_e) c_e \leq \frac{|S|^2}{2} c(\mathcal{T}).$$

□

Applying Lemma 3.1 to the optimum Steiner tree for T, we derive the following corollary.

Corollary 3.1. $f(v_T) \leq \frac{|T|}{2}\, ST(T)$.

Lemma 3.2. $\beta - 1 \leq \frac{f(v^*)}{|T| ST(T)}$.

Proof. This follows simply from the definition of $\beta = 1 + d(v^*, T)/ST(T)$ and the fact that $d(v^*, T) \leq \frac{1}{|T|}\, f(v^*)$. □

Corollary 3.2. $\beta \leq 1 + \frac{1}{2\alpha}$. *In particular,* $\beta \leq \frac{3}{2}$.

Proof. From Lemma 3.2, we have

$$\beta \leq 1 + \frac{f(v^*)}{|T|\, ST(T)} = 1 + \frac{f(v_T)}{\alpha |T|\, ST(T)} \leq 1 + \frac{1}{2\alpha},$$

the equality following from the definition of α and the last inequality from Corollary 3.1. □

Lemma 3.3. $\alpha \leq 2 - \frac{2}{|T|}$.

Proof. Let $t^* = \arg\min_{t \in T} d(v^*, t)$. Hence $f(v^*) \geq |T| d(v^*, t^*)$ and

$$\begin{aligned} f(v_T) &\leq f(t^*) = \sum_{t \in T \setminus \{t^*\}} d(t^*, t) \leq \sum_{t \in T \setminus \{t^*\}} (d(t^*, v^*) + d(v^*, t)) \\ &= (|T| - 1) d(t^*, v^*) + \sum_{t \in T \setminus \{t^*\}} d(v^*, t) \\ &= (|T| - 2) d(t^*, v^*) + f(v^*) \leq f(v^*) \left(2 - \frac{2}{|T|}\right). \end{aligned}$$

□

The following examples show that inequalities $\alpha \leq 2$ and $\beta \leq \frac{3}{2}$ are asymptotically tight. We adopt the following convention in Figures 3.2 and 3.5 : terminals are denoted by squares and the value k inside some indicates that k terminals lie at the same position. The black node represents the 1-median v^* of the terminals. The value close to an edge represents its cost.

Example 3.1. *Consider the instance depicted in Figure 3.2(a) where the terminals lie uniformly on a path of length $|T| - 1$ and are at distance 1 from v^*. Hence $f(v^*) = |T|$, $f(v_T) = 2|T| - 4$ (where v_T can be any terminal except the leftmost and rightmost ones), $ST(T) = |T| - 1$ and $ST(T \cup \{v^*\}) = |T|$. Therefore $(\alpha, \beta) = (2 - \frac{4}{|T|}, 1 + \frac{1}{|T|-1})$, which tends to $(2, 1)$ when $|T|$ tends to ∞.*

Example 3.2. *In Figure 3.2(b), $|T| = 2k + 1$ and the k leftmost and k rightmost terminals lie on a same node. One can verify that $f(v^*) = 2k - 1 - \epsilon$, $f(v_T) = 2k - 1$ (where v_T can be any left or rightmost terminal), $ST(T) = 2$ and $ST(T \cup \{v^*\}) = 3 - \frac{2}{k} - \epsilon$. When k tends to ∞ and ϵ tends to 0, (α, β) tends to $(1, \frac{3}{2})$.*

Corollary 3.2 shows that α must tend to 1 when β tends to $\frac{3}{2}$. Conversely it is shown in next section that β must tend to 1 when α tends to 2.

3.2.2 Better Inequalities Using Tree Partitions

For our derivation of the trade-off function, we first need to show the existence of partitions of any tree into a small number of trees, each of a given bounded size. This is stated in Proposition 3.1.

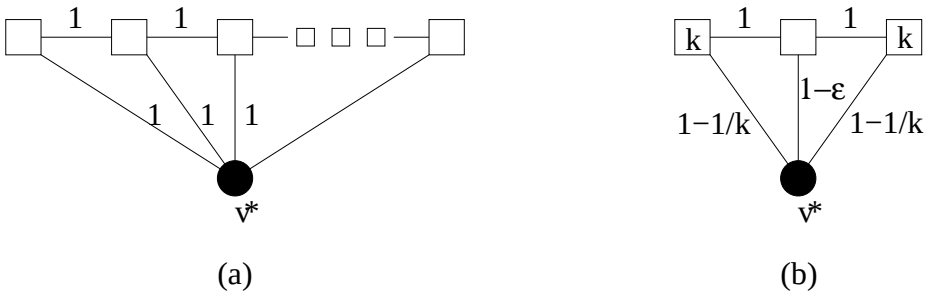

Figure 3.2: *Tight examples for $\alpha \leq 2$ and $\beta \leq \frac{3}{2}$.*

Definition 3.1. *A* (δ, M)-partition *of a weighted tree $\mathcal{T}$ is a collection of M subtrees $\mathcal{T}_i$ of $\mathcal{T}$ whose edge sets $E(\mathcal{T}_i)$ form a partition of E($\mathcal{T}$) and which satisfy $c(\mathcal{T}_i) \leq \delta c(\mathcal{T})$.*

Definition 3.2. *A graph H is a* subdivision *of a graph G if H is obtained after iterating the following operation on G: Introduce a new vertex w, pick an edge $e = \{u, v\}$ and replace it by two edges $f = \{u, w\}$ and $g = \{w, v\}$. For a weighted graph G, we impose that the previous operation be cost preserving,* i.e. *the cost of the new edges must satisfy $c_f + c_g = c_e$.*

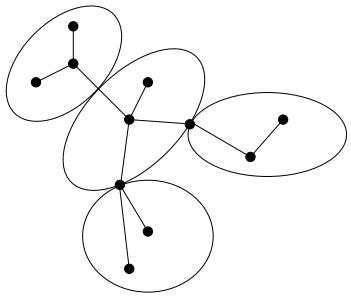

Figure 3.3: *A $(\delta, 4)$-partition of a subdivision of a tree.*

Proposition 3.1. *Let $\mathcal{T}$ be a tree. For all $0 < \delta \leq 1$, there exists a $(\delta, \lceil \frac{2}{\delta} \rceil - 1)$-partition of a subdivision of $\mathcal{T}$.*

An example of a (δ, M)-partition is shown in Figure 3.3. This result is tight in the sense that, for any positive integer M, the smallest value one can take for δ is indeed $\frac{2}{M+1}$ as comes out of Proposition 3.1. This is shown in the following example.

Example 3.3. *Consider a star with $M+1$ unit length spokes, as depicted in Figure 3.4(a). By the pigeonhole principle, in any (δ, M)-partition of a subdivision of the star, there exist two leaves lying in the same subtree. By connectivity, that subtree contains the path of length 2 between the two leaves. Hence $\delta \geq \frac{2}{M+1}$. Figure 3.4(b) illustrates a $\left(\frac{2}{M+1}, M\right)$ partition.*

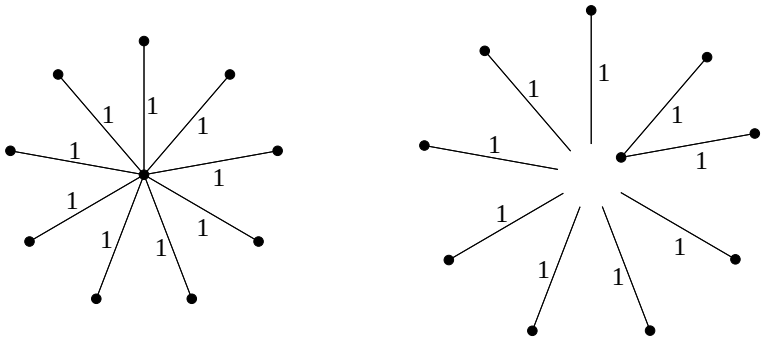

Figure 3.4: *Tight example for Proposition 3.1.*

Proof. We first prove the following claim by induction on n, the number of leaves of $\mathcal{T}$.

Claim 3.1. *For any value $C > 0$ and any leaf vertex v of $\mathcal{T}$, there exists a partition of a subdivision of $\mathcal{T}$ such that the subtree containing v has a cost at most $\frac{C}{2}$ and all other subtrees have a cost in $\left(\frac{C}{2}, C\right]$.*

Proof of the claim. The proof is by induction. The base case is $n = 2$. Then $\mathcal{T}$ is a path which we partition as follows: starting from the leaf opposite to v, cut pieces of cost C and remove them from the path until the remaining part has a cost less than C. If this cost is at most $\frac{C}{2}$ then the partition is done. Otherwise do the last cut by considering $\{v\}$ as a subtree (of cost 0). This partition fulfills the claim.

Suppose now that the claim is true for all trees with less than n leaves, where $n \geq 3$. Let w be the closest vertex to v of degree at least 3. If we cut $\mathcal{T}$ in w, we obtain the following connected components: the (v, w)-path and some subtrees $\mathcal{T}_i$ with less than n leaves. For each i, we construct by induction a partition of $\mathcal{T}_i$ that fulfills the condition of the claim with w chosen as the particular leaf. Since the subtrees of the $\mathcal{T}_i$ containing w

have cost at most $\frac{C}{2}$, we can merge them into bigger trees such that they all have a cost in $\left(\frac{C}{2}, C\right]$ except for one of them which has a cost at most $\frac{C}{2}$. If needed, consider that $\{w\}$ is that tree. Finally we extend the low cost tree on the $(w, v)-$path and finish the construction on this path as in the base case. □

Now consider the partition given by the previous claim with $C = \delta c(\mathcal{T})$. Consider the subtree of cost at most $\frac{\delta}{2}c(\mathcal{T})$ and another subtree adjacent to it. If the sum of their cost is less than $C = \delta c(\mathcal{T})$, merge them. In that case all the subtrees have cost in $\left(\frac{C}{2}, C\right]$. Otherwise remark that the average of their cost is in $\left(\frac{C}{2}, C\right]$. Let M be the number of subtrees obtained by the previous construction. We must have $M\frac{\delta}{2}c(\mathcal{T}) < c(\mathcal{T})$. Since M is integer, $M \leq \left\lceil \frac{2}{\delta} \right\rceil - 1$, as desired. □

From any (δ, M)-partition of the optimum Steiner tree, we can derive an inequality linking α and β as shown in the Proposition below. The idea of the proof is to bound the sum of the distances from any terminal t to the other terminals by going through the optimum Steiner tree for those terminals within the same member of the (δ, M)-partition as t is and going through v^* for the other terminals.

Proposition 3.2. *If $\beta > 1$ and if there exists a (δ, M)-partition of a subdivision of the optimum Steiner tree $\mathcal{ST}$ on T for some δ and M, then*

$$\alpha \leq 2 - \frac{2}{M} + \frac{\delta}{2(\beta - 1)M}.$$

Proof. If $M = 1$, then $\delta \geq 1$ and it suffices to prove $\alpha \leq \frac{1}{2(\beta-1)}$, which is equivalent to Corollary 3.2. If $\delta \geq 4(\beta - 1)$, it suffices to prove $\alpha \leq 2$, which is true because of Lemma 3.3.

So we can suppose w.l.o.g. that $M \geq 2$ and $\delta < 4(\beta - 1)$. Let us define $\delta = \mu(\beta - 1)$ with $0 \leq \mu < 4$.

Let $(\mathcal{T}_i)_{i=1\ldots M}$ be the subtrees of $\mathcal{T}$ in the (δ, M)-partition. These subtrees naturally induces a partition of T in M blocks $(B_i)_{i=1\ldots M}$ such that $B_i \subseteq V(\mathcal{T}_i)$. If a terminal $t \in T$ belongs to several subtrees, we arbitrarily choose its block.

For all i, we have

$$\begin{aligned}
|B_i|\, f(v_T) &\leq \sum_{t\in B_i} f(t) = \sum_{t\in B_i} f_{T\setminus B_i}(t) + \sum_{t\in B_i} f_{B_i}(t) \\
&\leq \sum_{t\in B_i}\sum_{t'\in T\setminus B_i} d(t,t') + \frac{|B_i|^2}{2} c(\mathcal{T}_i) \qquad \text{by Lemma 1} \\
&\leq \sum_{t\in B_i}\sum_{t'\in T\setminus B_i} \big(d(t,v^*) + d(v^*,t')\big) + \frac{|B_i|^2}{2}\delta\, ST(T) \\
&= |T\setminus B_i|\, f_{B_i}(v^*) + |B_i|\, f_{T\setminus B_i}(v^*) + \frac{|B_i|^2}{2}\delta\, ST(T) \\
&= (|T| - 2|B_i|)\, f_{B_i}(v^*) + |B_i|\, f(v^*) + \frac{|B_i|^2}{2}\delta\, ST(T).
\end{aligned}$$

We now consider two cases.

Case 1. If $|B_i| < \frac{|T|}{2}$ for all i, we use Lemma 3.2 and rewrite the last inequality as :

$$|B_i|\, f(v_T) \leq (|T|-2|B_i|)\, f_{B_i}(v^*) + |B_i|\, f(v^*) + \frac{|B_i|^2}{2}\frac{\delta}{(\beta-1)|T|}\, f(v^*).$$

A weighted sum over all i yields

$$\begin{aligned}
&f(v_T)\sum_{i=1}^{M}\frac{|B_i|}{|T|-2|B_i|} \\
&\qquad \leq f(v^*)\left(1 + \sum_{i=1}^{M}\frac{|B_i|}{|T|-2|B_i|} + \frac{\mu}{2|T|}\sum_{i=1}^{M}\frac{|B_i|^2}{|T|-2|B_i|}\right),
\end{aligned}$$

where $\mu = \frac{\delta}{\beta-1}$. Note that $0 < \mu < 4$. Let us do the following variable substitution: $z_i = \frac{|T|-2|B_i|}{|T|}$ or, equivalently, $|B_i| = |T|\frac{1-z_i}{2}$. Thus the z_i's satisfy

$$0 < z_i \leq 1 \text{ and } \sum_{i=1}^{M} z_i = M - 2.$$

Therefore,

$$\begin{aligned}
\alpha = \frac{f(v_T)}{f(v^*)} &\leq \frac{1 + \sum_{i=1}^{M} \frac{1-z_i}{2z_i} + \frac{\mu}{8} \sum_{i=1}^{M} \frac{(1-z_i)^2}{z_i}}{\sum_{i=1}^{M} \frac{1-z_i}{2z_i}} \\
&= 1 + \frac{1 + \frac{\mu}{8}(\sum_{i=1}^{M} \frac{1}{z_i} - 2M + M - 2)}{\frac{1}{2}\sum_{i=1}^{M} \frac{1}{z_i} - \frac{1}{2}M} \\
&= 1 + \frac{8 + \mu(Z - M - 2)}{4(Z - M)} \quad \text{where } Z = \sum_{i=1}^{M} \frac{1}{z_i} \\
&= 1 + \frac{\mu}{4} + \frac{4 - \mu}{2(Z - M)}.
\end{aligned}$$

Since $\mu < 4$, the last term is maximized when Z is minimum. By the Cauchy-Schwarz inequality,

$$M^2 \leq \left(\sum_{i=1}^{M} z_i\right)\left(\sum_{i=1}^{M} \frac{1}{z_i}\right) = (M-2)Z.$$

Hence

$$\alpha \leq 1 + \frac{\mu}{4} + \frac{(4-\mu)(M-2)}{4M} = 2 - \frac{2 - \mu/2}{M},$$

proving the desired bound.

Case 2. If $|B_i| \geq \frac{|T|}{2}$ for some i, then for this i, we have

$$\begin{aligned}
|B_i|\, f(v_T) &\leq (|T| - 2|B_i|)\, f_{B_i}(v^*) + |B_i|\, f(v^*) + \frac{|B_i|^2}{2}\delta\, ST(T) \\
&\leq (|T| - 2|B_i|)|B_i|(\beta - 1)\, ST(T) + |B_i|\, f(v^*) \\
&\quad + \frac{|B_i|^2}{2}\mu(\beta - 1)\, ST(T)
\end{aligned}$$

using $\frac{f_{B_i}}{|B_i|} \geq d(v^*, T) = (\beta - 1)ST(T)$, as in the proof of Lemma 3.2. Hence,

$$f(v_T) \leq f(v^*) + \left(1 - 2\frac{|B_i|}{|T|} + \frac{\mu}{2}\frac{|B_i|}{|T|}\right)|T|(\beta - 1)\, ST(T).$$

Since $f(v_T) \geq f(v^*)$, $1 - \left(2 - \frac{\mu}{2}\right)\frac{|B_i|}{|T|} \geq 0$. Using Lemma 3.2, $\mu < 4$ and $M \geq 2$, we have

$$\alpha \leq 2 - \left(2 - \frac{\mu}{2}\right)\frac{|B_i|}{|T|} \leq 2 - \left(2 - \frac{\mu}{2}\right)\frac{1}{2} \leq 2 - \frac{2 - \mu/2}{M},$$

completing the proof.

□

We can now prove the following inequalities:

Theorem 3.1. *If $\beta > 1$ then, for any positive integer M,*

$$\alpha \leq 2 - \frac{2}{M} + \frac{1}{M(M+1)(\beta-1)}. \tag{3.1}$$

In particular,

$$\alpha \leq 3 - \frac{1}{2-\beta}. \tag{3.2}$$

The lower envelope of all inequalities (3.1) corresponds to the dotted line in Figure 3.1, while (3.2) corresponds to the continuous line in the Figure.

Proof. For each positive integer M, there exists a (δ, M)-partition of the optimum Steiner tree $\mathcal{ST}$ with $\delta = \frac{2}{M+1}$ by Proposition 3.1. Proposition 3.2 immediately yields inequality (3.1).

Consider two inequalities (3.1) with $M = n$ and $M = n+1$ where n is a positive integer. The first inequality is stronger than the second one if and only if:

$$2 - \frac{2}{n} + \frac{1}{n(n+1)(\beta-1)} \leq 2 - \frac{2}{n+1} + \frac{1}{(n+1)(n+2)(\beta-1)},$$

or equivalently $\beta \geq 1 + \frac{1}{n+2}$. Therefore each inequality (3.1) is the strongest one when $\beta \in \left[1 + \frac{1}{M+2},\ 1 + \frac{1}{M+1}\right]$, which corresponds to $\alpha \in \left[2 - \frac{1}{M}, 2 - \frac{1}{M+1}\right]$. By eliminating the parameter M, one can easily see that all points $(\alpha, \beta) = (2 - \frac{1}{M}, 1 + \frac{1}{M+1})$ lie on the curve $\alpha = 3 - \frac{1}{2-\beta}$. Observe finally that the right-hand side of inequality (3.1) is convex in β and the right-hand side of inequality (3.2) is concave in β. This proves the validity of inequality (3.2). □

The inequalities of Theorem 3.1 are not likely to be tight. The worst examples we know are the following.

Example 3.4. *In the instance depicted in Figure 3.5, there are M groups of k terminals plus another $M-1$ terminals between the groups, for a total of $kM + M - 1$ terminals. When k tends to ∞, this can be seen to correspond to $(\alpha, \beta) = (2 - \frac{2}{M}, 1 + \frac{1}{2M-2})$. E.g., for $M = 3$, this gives $(\alpha, \beta) = (\frac{4}{3}, \frac{5}{4})$.*

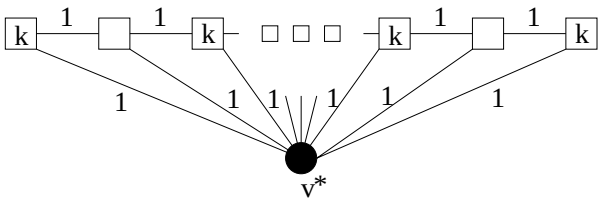

Figure 3.5: *Worst known examples.*

3.3 Application to the Minimum Bandwidth Cost Core Location Problem

As discussed in chapter 2, the *Minimum Bandwidth Cost Core Location Problem* is the problem of finding a vertex v which minimizes $cost(v) = f(v) + \lambda\, ST(T \cup \{v\})$.

If we use a particular ρ-approximation algorithm to compute the Steiner tree costs in a core enumeration algorithm, we clearly obtain a ρ-approximation algorithm for the core location problem as well, independently of the value of λ. ([50]). However we prove in this section that we can approximate the Minimum Bandwidth Cost Core Location Problem with a ratio close to ρ by computing only one ρ-approximation of a minimum cost Steiner tree. Namely we consider the following core location algorithm:

CoreLocation
input: A graph $G = (V, E, c)$ and a set of terminals $T \subseteq V$.
output: A core node.

1. Compute a Steiner tree $\mathcal{ST}_\rho$ spanning T with a ρ-approximation algorithm.
2. Compute $v^* = \arg\min_{v \in V} f(v)$ and $\overline{cost}(v^*) = f(v^*) + \lambda(c(\mathcal{ST}_\rho) + d(v^*, T))$.
3. Compute $v_T = \arg\min_{t \in T} f(t)$ and $\overline{cost}(v_T) = f(v_T) + \lambda c(\mathcal{ST}_\rho)$.
4. **If** $\overline{cost}(v^*) \leq \overline{cost}(v_T)$, **then return** v^*, **else return** v_T.

The main advantage of the algorithm is that we need to solve only one Steiner tree instance. Indeed, we already remarked that, in a real setting, finding a good Steiner tree is typically the bottleneck in terms of computation and communication time, whereas the evaluation of f is fast since the

shortest path distances are usually pre-computed and stored in routing tables.

Proposition 3.3. CoreLocation *is a* $\left(1+\frac{\rho^2}{4}\right)$*-approximation algorithm for the Minimum Bandwidth Cost Core Location Problem.*

Proof. CoreLocation returns a node v^{CL} with

$$\begin{aligned}\overline{cost}(v^{CL}) &= \min\left\{f(v^*)+\lambda(c(\mathcal{ST}_\rho)+d(v^*,T))\,,\; f(v_T)+\lambda c(\mathcal{ST}_\rho)\right\}\\ &\leq \min\left\{f(v^*)+\lambda(\rho\, ST(T)+d(v^*,T))\,,\; f(v_T)+\lambda\rho\, ST(T))\right\}\\ &= \min\left\{f(v^*)+\lambda(\rho+\beta-1)\, ST(T)\,,\; \alpha\, f(v^*)+\lambda\rho\, ST(T)\right\}.\end{aligned}$$

Clearly a lower bound on the minimum cost is given by:

$$\min_{v\in V} cost(v) \geq f(v^*)+\lambda ST(T).$$

Therefore, if we set

$$\mu = \frac{f(v^*)}{f(v^*)+\lambda\, ST(T)},$$

for fixed value of α and β, the approximation ratio of *CoreLocation* is upper bounded as follows:

$$\frac{\overline{cost}(v^{CL})}{\min_{v\in V} cost(v)} \leq \max_{0\leq\mu\leq 1} \min\{\mu+(1-\mu)(\rho+\beta-1)\,,\; \mu\alpha+(1-\mu)\rho\}. \quad (3.3)$$

Remark that, if $\rho \geq \alpha$, then the second argument of the minimum in (3.3) is at most $\rho \leq 1+\frac{\rho^2}{4}$. Thus we can suppose w.l.o.g. that $\rho < \alpha$. Hence the first (*resp.* second) argument is a non-increasing (*resp.* non-decreasing) linear function in μ. Also the first argument is at least (*resp.* at most) the second one in $\mu = 0$ (*resp.* $\mu = 1$). This implies that the maximum is attained when μ satisfies

$$\begin{aligned}& \mu+(1-\mu)(\rho+\beta-1) = \mu\alpha+(1-\mu)\rho\\ \Leftrightarrow\quad & \mu = \frac{\beta-1}{\alpha+\beta-2}.\end{aligned}$$

The upper bound (3.3) becomes

$$\frac{\overline{cost}(v^{CL})}{\min_{v\in V} cost(v)} \leq \frac{\alpha\beta+\alpha(\rho-1)-\rho}{\alpha+\beta-2} = \alpha+\frac{(\alpha-1)(\rho-\alpha)}{\alpha+\beta-2}. \quad (3.4)$$

Observe that the right-hand side of inequality (3.4) is a non-decreasing function in β (for $\beta \geq 1$) since $\alpha \geq 1$ and $\rho < \alpha$. Therefore, by Theorem 3.1, we can upper bound the approximation ratio by maximizing the right-hand side of inequality (3.4) along the curve $\alpha = 3 - \frac{1}{2-\beta}$. However, to make this maximization easier, we will first derive a new inequality :

$$\alpha \leq 3 - \frac{1}{2-\beta} = 3 - \frac{1}{1-(\beta-1)} \leq 3 - (1 + (\beta - 1)) = 3 - \beta.$$

If we maximize the right-hand side of inequality (3.4) along the line $\alpha+\beta = 3$, the upper bound becomes :

$$\begin{aligned} \frac{\overline{cost}(v^{CL})}{\min_{v \in V} cost(v)} &\leq \max_{1 \leq \alpha \leq 2} \alpha + (\alpha - 1)(\rho - \alpha) \\ &= \max_{1 \leq \alpha \leq 2} \alpha(2 + \rho - \alpha) - \rho \\ &= \left(1 + \frac{\rho}{2}\right)^2 - \rho = 1 + \frac{\rho^2}{4}, \end{aligned}$$

where the maximum is attained for $\alpha = 1 + \frac{\rho}{2}$.

□

Let us look at some particular approximation algorithms for the Minimum Cost Steiner Tree problem. If we use a minimum spanning tree algorithm, $\rho = 2$ (see [62]). Hence, by Proposition 3.3, we obtain a 2-approximation algorithm as well for the core location problem. So in this case it is impossible to improve the approximation ratio of *CoreLocation* by computing other Steiner trees. The minimum spanning tree problem has also the main advantage to have distributed algorithms (see [27] for instance) which are therefore easier to implement in a real network. If we use the best known approximation algorithm for the Steiner tree problem ($\rho \approx 1.55$), then we obtain a ratio of 1.60. Finally if we could always compute the optimal Steiner tree spanning the terminals, *CoreLocation* would have an approximation ratio of 1.25. Note that, by using the inequalities of Theorem 3.1 instead of $\alpha + \beta \leq 3$, one can actually prove that *CoreLocation* is a 1.2-approximation algorithm when $\rho = 1$ (This is attained for $\alpha = 3/2$ and $\beta = 4/3$ in inequality (3.4)).

3.4 Conclusion

In this chapter, we have proved the existence of a core node with a 1-median and a Steiner tree criteria simultaneously small according to the

trade-off function of Theorem 3.1. In particular one can always find such a core node among the terminals or the 1-median. These results have allowed us to prove that, given a Steiner tree algorithm with a known performance guarantee, computing only one Steiner tree approximation instead of considering all potential core locations leads only to a small increase of the guaranteed approximation ratio for the Minimum Bandwidth Cost Core Location Problem. In a real setting, a core location protocol would therefore naturally trade off a small loss of routing performance against a gain in computation time.

Part II

Approximation of Forward Error Correction Robustness with the Peakedness Characterization

Chapter 4

Increasing Forward Error Correction Robustness with Multipath Routing

As discussed in Chapter 1, retransmission mechanisms are widely used in the Internet to protect applications from packet losses. However they are usually not suitable for some delay-sensitive multimedia applications, which prefer the flexibility of Forward Error Correction (FEC). We recall that a (N, K)- FEC block code consists of partitioning the packet stream into blocks of K consecutive packets and in adding $R = N - K$ redundant data packets to each block. The required property of the coding technique is that, for each N-packet block, if at least K packets are transmitted successfully, the receiver is able to recover the K original data packets. Otherwise some data is definitely lost and we say that the block has incurred an *irrecoverable loss*. In practice this property is achieved by using Reed-Solomon erasure codes ([49]).

The robustness of such a protection scheme is thus measured by the probability of irrecoverable loss (PIL) within an arbitrary block. For a given block, if X denotes the number of packets successfully transmitted and is viewed as a random variable, the PIL can be written as a function P_{IL} of the FEC parameters (R, K) as follows :

$$P_{IL}(R, K) = \mathsf{P}[X < K] = \mathsf{P}[N - X > R]. \tag{4.1}$$

Remark that we see P_{IL} as a function of R and K instead of N and K in order to highlight the symmetry between R and K in equation (4.1). This symmetry should become more obvious throughout the following

sections. We must also point out that, in equation (4.1), many parameters needed to compute P_{IL} are actually hidden. In particular, a loss model must be given in order to compute the probability distribution of X.

In simple models, the transmission channel is characterized by a single parameter p giving the average packet loss rate. If packet losses are assumed to be mutually independent events, then X follows a binomial distribution and therefore,

$$P_{IL}(R,K) = \sum_{x=0}^{K-1} \left(\begin{array}{c} R+K \\ x \end{array} \right) (1-p)^x p^{R+K-x}.$$

In this case the FEC protection is efficient when the parameters are chosen in such a way that the expected number of lost packets within a block, $p\,N = p\,(R+K)$, is significantly smaller than the number of redundant packets R. Equivalently, the *redundancy ratio* $\frac{R}{N}$ must be significantly greater than the average loss rate p.

However, in communication networks, transmission errors typically arise in bursts. In the Internet, router congestion is to a large extent responsible for this behavior. Each router has indeed a finite buffer containing packets to be forwarded. When this buffer is full, the router has to drop packets upon each new packet arrival. This is usually done according to a *Drop Tail* policy, i.e. only the last packets to arrive are discarded. Consequently the packet loss events cannot be assumed to be independent and thus these losses are not evenly distributed over time. The same behavior may be observed with other types of network technologies, such as wireless networks, where fading and shadowing may cause high error rates concentrated in failure period. FEC techniques then become inefficient when the typical error burst size experienced by a data stream is greater than R, even if the average loss ratio is much smaller than the redundancy ratio $\frac{R}{N}$.

In this chapter and the next, our main goal is to maximize the robustness of FEC, i.e. to minimize the PIL of a block. An obvious way of achieving our goal would be to choose sufficiently large values of N and R in such a way that $\frac{R}{N}$ and R are significantly larger than the average loss rate and the expected number of packets lost consecutively, respectively. However we cannot design coding schemes with arbitrarily large values for R and N. A large value for $\frac{R}{N}$ would indeed add too much redundancy into the data flow, leading to a waste of bandwidth and an increase of congestion risk, while a large N would increase the coding and decoding time and therefore the global transmission delay. Hence this would also de-

crease the benefits of FEC for delay-sensitive applications. Throughout this chapter, we actually assume that FEC parameters have been chosen once and for all, so as to overcome these tradeoffs.

We must also point out that other criteria than the PIL could a priori be considered as measures of robustness. Indeed, the possibility of decoding FEC blocks beyond the error-correction bound have been studied e.g. in [56] for Reed-Solomon codes and in [16] in the context of image transmissions. Moreover one could argue that, even if an irrecoverable loss occurs, the received packets may contain useful data anyway. Therefore minimizing the expected number of packets lost beyond the bound R could also be considered as an objective. However this argument is not valid for some applications. For instance, when protecting video transmissions, several video frames are typically encoded within a FEC block in such a way that the video data of each frame is spread over the first K packets of the block, as depicted in figure 4.1. In this case, when less than K packets are correctly transmitted, all the frames of the block are corrupted and therefore the data received may be considered as useless. This protection scheme is used for instance in [55] for unequal error protection with video streaming. Consequently, in this thesis, we will use the PIL as the only measure of robustness for FEC techniques.

Recently it has been proposed to increase the robustness of FEC for media streaming applications on the Internet by splitting the transmission of a FEC block onto several paths (see [37], [46],[47],[48]). Intuitively this increases the transmission time between consecutive packets sent on the same path, hence decreasing the dependency between packet losses. Alternatively the benefits of multipath routing may be explained by the following self-evident fact : for a given failure period on a path and a given packet sending rate, less packets of a block are lost if only a fraction of the block is sent on this path than if all packets of the block are sent on the same path. More generally, it is worth noting that the use of path diversity to improve multimedia application support on IP networks is a very active area of research in the networking literature. A review of multipath routing benefits is given in [3].

This chapter is organized as follows. In Section 4.1, we review the model which has been used to show that multipath routing can increase FEC robustness. The approach proposed so far uses a Markovian loss model and relies on an exact computation of the PIL. Hence it is limited to very small network instances. In order to extend this model to larger network instances, several modeling issues need to be discussed. Moreover,

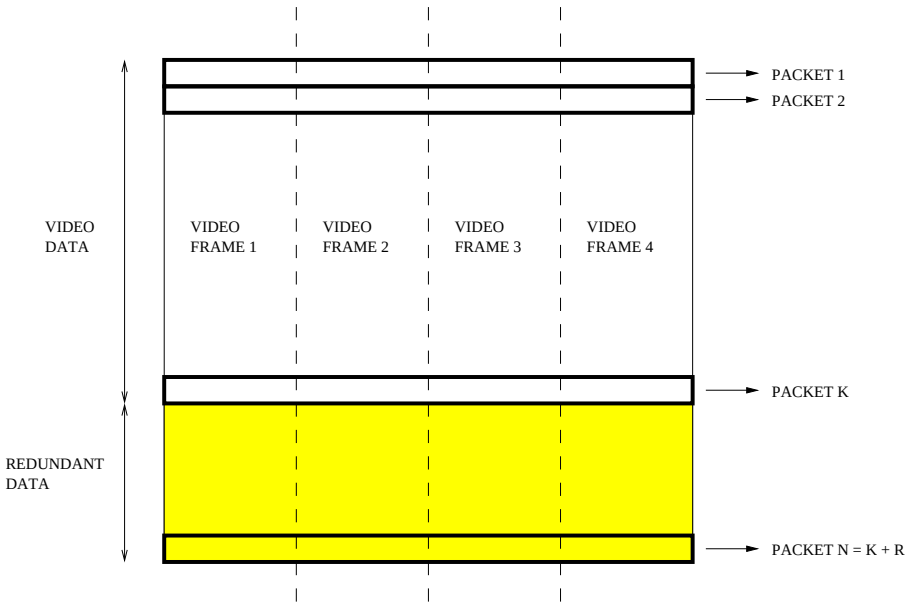

Figure 4.1: *Example of protection scheme for video frames with Forward Error Correction*

from a computational perspective, existing approaches are difficult to generalize. Indeed the exact computation of the PIL becomes intractable for large network instances. In particular numerical examples are given and used to motivate the usefulness of a traffic characterization, called *peakedness*. The peakedness of a flow may be seen as a measure of its "burstiness", that is as a measure of how packets are spread over time within the flow. In Section 4.2 we present this notion of peakedness and we propose a method which approximates the PIL of a FEC block by looking at the impact of network losses on the packet flow peakedness. The method is devised so as to scale well with the network size. We would like to mention that, while the peakedness notion is a well-known tool used in the teletraffic literature to model telephone calls arrival processes (see [11] or [24] for example), the question of its usefulness to model data flows on the Internet remains largely uninvestigated so far.

4.1 Path Diversity Models for Multimedia Streaming

In this section, our starting point is the work of Nguyen and Zakhor ([46], [47], [48]) which shows how multipath routing can increase the robustness of FEC techniques. Our goal is to present their modeling assumptions and to discuss what are their shortcomings if one wants to tackle larger network instances than they do. We address in turn modeling issues concerning the network, the packet loss processes and the traffic in Sections 4.1.1, 4.1.2 and 4.1.3. Finally in Section 4.1.4, we address the problem of computing the PIL of a FEC block for simple network instances.

4.1.1 Network Model

Throughout this chapter and the next, we model the network as a directed graph $G = (V, A)$. It is worth noting that the graph considered is not intended to model the complete Internet topology. As discussed in Chapter 1, we prefer instead to view the graph G as an overlay network. Indeed, the deployment of scalable multimedia streaming applications on the Internet is often achieved by a Content Delivery Network (CDN). A CDN can be seen as an overlay network lying at the application level and whose nodes correspond to the servers owned by a particular company and to the clients requesting access to some content. Akamai is often cited as an example of a CDN company that strategically owns servers all around the Internet and controls data forwarding and replication between them in order to achieve load balancing, low latency and high throughput. As already mentioned, they have designed a multi-tier architecture aimed at live video streaming. The related routing optimization on their overlay network is discussed in [2].

In such an overlay network, there is an arc for any pair of nodes which directly communicate via IP routing. Such an arc is sometimes called an IP *tunnel*. The advantage of seeing G as an overlay architecture is that, given IP tunnels between nodes of interest, we can assume complete control of the routing at the overlay network level. A potential drawback of this approach is the fact that distinct arcs of the overlay network may actually be paths having links in common at the IP level. This would prevent us from assuming that arc characteristics are mutually independent. However throughout this work we assume that this effect can be neglected. Note that, in practice, our assumption could be easily validated by using the `traceroute` IP utility which directly lists the IP addresses of routers used to forward data packets from one node to another.

In summary, we thus assume that the vertex set V corresponds to points or stations in the network where we have control of the forwarding process of the data flows, and that each arc (i, j) in A actually represents a whole path (at the IP layer) on which data is routed between vertices i and j. We must point out that, despite this general description of the network model, Nguyen and Zakhor's results focus on graphs consisting of two vertices linked by parallel arcs. As shown in Section 4.1.4, these simple networks suffice to exhibit the advantage of multipath routing for increasing FEC robustness. We now describe how to characterize losses on arcs.

4.1.2 Loss Model

Packet losses in the Internet are mostly due to a congestion state in the packet buffer of a router. If a transmission path goes through a congested buffer, this path is likely to experience a high packet loss rate for a short amount of time. This network behavior explains to some extent why, when looking at Internet traffic traces, packet losses generally occur in bursts rather than being uniformly spread over time. At first glance, a natural loss model would therefore explicitly consider finite packet queues at the tail of each arc. However, for the purpose of computing the probability of irrecoverable loss of FEC blocks, we simply want a model which describes *how* packet losses occur, i.e. which essentially characterizes the frequency and the lengths of error bursts. Therefore our model does not need to include packet queues even if they physically explains *why* packet losses occur in bursts.

The characterization of packet losses within the network is indeed a key feature of models used in [37], [47] and [48]. To take into account the interdependency of consecutive packet loss events on transmission paths, they use one of the simplest models that one can think of, known as the *Gilbert loss model*. In this model, losses on a transmission path $a \in A$ are characterized by a 2-state Markov process (see [26] or [51]) $\{M^{(a)}(t), t \in \mathbb{R}\}$, where $M^{(a)}(t) \in \{0, 1\}$ for all $t \in \mathbb{R}$. The process states must be interpreted as follows : at time t, if a packet is transmitted on a and $M^{(a)}(t) = 0$ then the packet is lost , whereas it is successfully transmitted if $M^{(a)}(t) = 1$. Each process is characterized by transition rates $\mu_0^{(a)}$ from state 0 to 1 and $\mu_1^{(a)}$ from state 1 to 0, as depicted in figure 4.2. For $i \in \{0, 1\}$, we denote the stationary probability of state i by $\pi_i^{(a)}$. These

stationary probabilities are then given by :

$$\pi_0^{(a)} = \frac{\mu_1^{(a)}}{\mu_0^{(a)} + \mu_1^{(a)}}, \tag{4.2}$$

$$\pi_1^{(a)} = \frac{\mu_0^{(a)}}{\mu_0^{(a)} + \mu_1^{(a)}}. \tag{4.3}$$

Every Markov process $M^{(a)}$ is assumed to be in stationary state. Hence, $\mathsf{P}[M^{(a)}(t) = i] = \pi_i^{(a)}$ for any $t \in \mathbb{R}$. Moreover, according to our assumption of independence between arcs, we also suppose that all the Markov processes considered are mutually independent. In the remainder of this thesis, such a Markov process will be called a $(\mu_0^{(a)}, \mu_1^{(a)})$-*Markov process*. We will always use the notation $\pi_0^{(a)}$ and $\pi_1^{(a)}$ for its stationary probabilities.

Note that, from now on, we omit to specify superscript a when it is useless to specify the arc with which the Markov process is associated.

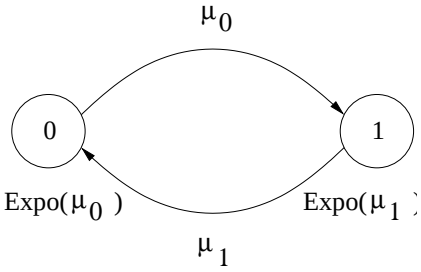

Figure 4.2: *Markov process characterizing losses on an arc*

Transition densities (μ_0, μ_1) may conceptually be interpreted in several ways :

1. The process stays in state $i \in \{0, 1\}$ for a random time that is exponentially distributed of parameter μ_i and then switches to state $1 - i$. In particular, the expected amount of time spent in state i before a transition is $\frac{1}{\mu_i}$
2. The process is driven by a Poisson process of arrival rate $\mu_0 + \mu_1$. At each arrival of this Poisson process, the Markov process stays or switches - depending on its previous state - to state i with probability π_i, $i \in \{0, 1\}$. In the literature, this description of the process is usually called *uniformization* of the Markov process.

3. Finally, by looking at the states after consecutive small time increments δ, one can turn the Markov process into a *sampled time Markov chain* with transition probabilities $p_{i,1-i} = \mu_i\delta + o(\delta)$ and $p_{i,i} = 1 - \mu_i\delta + o(\delta)$ (see figure 4.3).

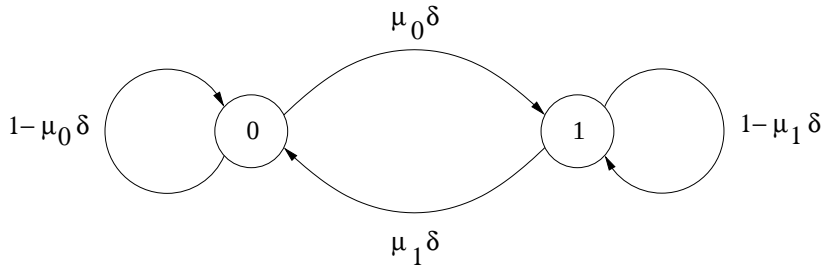

Figure 4.3: *Equivalent Markov chain obtained by considering states after infinitesimal time increments*

In a more detailed model, parameters (μ_0, μ_1) could be a function of the link load to capture the fact that congestion causes more losses. But for the sake of simplicity, we will assume that, for all arcs, transition densities are constant. From a practical point of view, this is somehow equivalent to neglecting the impact of the routing of data flows on the network congestion state.

Remark that π_0 can be seen as the average loss rate of the arc and $\frac{1}{\mu_0}$ as the average length of a failure period. Therefore, if we multiply μ_0 and μ_1 by common factor γ, π_0 stays constant while $\frac{1}{\mu_0}$ is divided by γ. This loss model thus captures the fact that two transmission paths may at the same time have similar average loss rate but very different loss characteristics. Besides, despite its apparent simplicity, the Gilbert model has been shown to approximate loss behavior on the Internet quite accurately ([7], [31], [64]). In [64] for instance, the accuracy of the model is assessed by a statistical parameter estimation from long Internet traffic traces. In their numerical characterization, Nguyen and Zakhor consider realistic values of the Markov parameters μ_0 and μ_1 according to results reported in [64] and to their own experiments.

Finally we must point out that modeling and analyzing the variability of failure times is also an issue of concern in other contexts than communication networks. For instance, they exists a broad literature on the origins and consequences of this variability in production systems (see [53] for instance). Whereas the variability is also related to congestion effects in

this context, the main issue is to analyze its effect on the global efficiency of production lines. Therefore it is a priori not clear whether this literature might yield alternative models of variable packet loss or other ways of analyzing the efficiency of FEC.

4.1.3 Packet Traffic Modeling Issues

Since we have introduced a notion of time in the network model, packet streams at a given point in the network cannot be simply modeled by their average flow rate. In order to potentially include a very large class of traffic patterns in our discussion, a packet stream at a given node is modeled as a *point process* ([15]). A point process is described by a real-valued random sequence of arrival times $\{T_n,\, n \in \mathbb{Z}\}$. In our case, T_n must be interpreted as the arrival time of the n^{th} packet at the node. Remark that we do not consider the temporal length of a packet but we simply model it as a point along the time axis. Also note that arrival times are indexed with integers in $\mathbb{Z}$ and not in $\mathbb{N}$ to highlight the fact that the process has been running for infinitely long time.

Let X denote a point process with arrival times $\{T_n, n \in \mathbb{Z}\}$. We now describe auxiliary objects typically associated with X.

- X can be alternatively characterized by its *counting process* $\{N_X(t), t \in \mathbb{R}\}$, where $N_X(t)$ is defined as the number of arrivals in the interval $(0, t]$ for $t \geq 0$. For $t \leq 0$, we pose $N_X(t) = -N_X(-t)$. In addition it is convenient to define, for all $t_1, t_2 \in \mathbb{R}$, $\widetilde{N}_X(t_1, t_2) = N_X(t_2) - N_X(t_1)$, which hence gives the number of arrivals in $(t_1, t_2]$ when $t_1 \leq t_2$.
- The sequence of *inter-arrival times* $\{A_n,\, n \in \mathbb{Z}\}$ is given by $A_n = T_n - T_{n-1}$.
- The value $\lambda_X = \frac{1}{\mathsf{E}[A_n]}$ is called the *arrival rate* of the process. Hence, is satisfies $\mathsf{E}[\widetilde{N}_X(t_1, t_2)] = \lambda_X(t_2 - t_1)$, for all $t_1, t_2 \in \mathbb{R}$.
- The *covariance density* of X, $\widetilde{k}_X(t_1, t_2)$, is defined as

$$\widetilde{k}_X(t_1, t_2) = \lim_{\Delta t \to \infty} \frac{\mathsf{Cov}\left[\widetilde{N}_X(t_1, t_1 + \Delta t), \widetilde{N}_X(t_2, t_2 + \Delta t)\right]}{\Delta t^2}. \quad (4.4)$$

  As shown in the examples below, the limit in equation (4.4) may not exist and the covariance density should therefore be formally

defined as the generalized function, or distribution, (see e.g. [28]) $\widetilde{k}_X$ which satisfies :

$$\mathsf{Cov}\left[\widetilde{N}_X(t_1, t_1+\Delta t_1), \widetilde{N}_X(t_2, t_2+\Delta t_2)\right] = \int_{t_1}^{t_1+\Delta t_1}\int_{t_2}^{t_2+\Delta t_2} \widetilde{k}_X(\tau_1,\tau_2)d\tau_2 d\tau_1, \tag{4.5}$$

for any $t_1, \Delta t_1, t_2, \Delta t_2 \in \mathbb{R}$.

Remark that λ_X and $\widetilde{k}_X$ allow us to compute essentially any expectation or covariance for the counting process of X. Consequently the pair $(\lambda_X, \widetilde{k}_X)$ is said to be a *second-order characterization* of X.

Since the loss processes introduced in Section 4.1.2 are stationary, we naturally assume that all the point processes used to model packet streams are stationary as well, i.e. they are invariant with respect to time translations. This assumption has some consequences for some of the objects introduced above.

- The sequence of inter-arrival times $\{A_n,\, n \in \mathbb{Z}\}$ becomes stationary. In particular, variables $A_n, n \in \mathbb{Z}$ are identically distributed (but are of course not independent). Note that, when variables A_n are also required to be mutually independent, then X is a *renewal process*.

- For a given $t \in \mathbb{R}$, variables $N_X(t)$ and $\widetilde{N}_X(\tau, \tau+t)$, $\tau \in \mathbb{R}$, are all identically distributed.

- The covariance density $\widetilde{k}_X(t_1, t_2)$ is now a function of $t_2 - t_1$. So we conveniently redefine the covariance density of a stationary point process X as the generalized function $k_X(t)$ which satisfies $k_X(t_2 - t_1) = \widetilde{k}(t_1, t_2)$.

Example 4.1. We give here the covarariance density of two simple point processes :

1. **Poisson process of rate λ.** For all $t \geq 0$, $N(t)$ is a discrete Poisson variable of parameter λt. In particular $\mathsf{Var}[N(t)] = \lambda t$. Because of the memoryless property of Poisson processes, the covariance of $\widetilde{N}_X(t_1, t_1+\Delta t_1)$ and $\widetilde{N}_X(t_2, t_2+\Delta t_2)$ is $\lambda\Delta t$ if intervals $(t_1, t_1+\Delta t_1]$ and $(t_2, t_2+\Delta t_2]$ overlap on an interval of length Δt. Therefore, the covariance density of the Poisson process is given by $k(t) = \lambda\delta(t)$, where $\delta(t)$ is the Dirac delta function.

2. **Deterministic point process of rate λ.** Here inter-arrival times are deterministic and equal $\tau = \frac{1}{\lambda}$. Consider two infinitesimal intervals $(t_1, t_1 + dt]$ and $(t_2, t_2 + dt]$. Then, for $i = 1, 2$, $\widetilde{N}(t_i, t_i + dt)$ is a Bernoulli variable and equals 1 with probability $\frac{dt}{\tau} = \lambda dt$. If $t_2 - t_1$ is a multiple of τ, then $\widetilde{N}(t_1, t_1 + dt)$ and $\widetilde{N}(t_2, t_2 + dt)$ are equal. Hence their covariance is $\lambda dt - (\lambda dt)^2$. Otherwise, $\widetilde{N}(t_1, t_1 + dt)$ and $\widetilde{N}(t_2, t_2 + dt)$ cannot equal 1 at the same time and, hence, their covariance is $-(\lambda dt)^2$. In summary this shows that the covariance density of the process is given by :

$$k(t) = -\lambda^2 + \lambda \sum_{n=-\infty}^{\infty} \delta(t - n\tau).$$

The nature of the packet streams at various point in the network naturally depends on the Markov loss parameters introduced in Section 4.1.2 but also on

1. the nature of the packet streams when they enter the network,
2. the way packets are routed on the network.

Packet Streams Entering the Network

Nguyen and Zakhor only consider a deterministic way of sending packets on the network. The incoming packet stream is thus completely characterized by its sending rate λ or equivalently by the packet inter-arrival time $\tau = \frac{1}{\lambda}$. However, as illustrated in Section 4.1.4, we can a priori consider other types of incoming packet stream, such as a Poisson process or any renewal processes.

Packet routing

One of the issues in the routing model is to specify how to split a given packet stream onto several paths. Even if we are given the average flow rate of each of the sub-streams, it does not univocally characterize their nature. We can think of at least two ways of splitting of packet stream :

1. **deterministic round-robin.** For instance, if the stream must be evenly split onto two paths, every other packet is forwarded on one of the paths. However, when the stream must be split unevenly

among several paths, it becomes more difficult to define a natural way of deterministically dividing the stream, even though some theoretical techniques, such as *billiard sequences*, have been proposed in the literature (see [59]).

2. **random splitting.** If the stream must be split in k sub-streams according to ratio $p_1 : p_2 : \ldots : p_k$, with $\sum_{i=1}^{k} p_i = 1$, then each packet is independently included in stream i with probability p_i. Therefore this splitting method is well defined for any ratio. Moreover it has a nice property when dealing with Poisson processes : if a Poisson process of rate λ is split according to ratio p_i, then the k sub-streams are independent Poisson processes, where sub-stream i has an arrival rate $\lambda_i = p_i\lambda$.

Nguyen and Zakhor deterministically split the packet stream. Actually they do not explicitly develop how they achieve an uneven split. We understand that they proceed as follows : to split a packet stream with deterministic inter-arrival time τ according to ratio $p_1 : p_2 : \ldots : p_k$, they consider that sub-stream i has deterministic inter-arrival time $\frac{\tau}{p_i}$. Note that this way of splitting the stream implies that some packets are delayed.

Another issue concerns the delay that packets incur when they travel across network links. This is particularly important when we merge previously split sub-streams. We must indeed know to what extent they were desynchronized by traveling through different paths. We can formulate several hypotheses :

1. **Independence**. We assume that the streams to be merged have incurred very different delays in such way that they can be considered independent. This is probably the most convenient hypothesis to work with. But we will not use it in this thesis.

2. **Re-synchronization.** We suppose that the delay is zero or, equivalently, that packets travel at infinite speed. Hence when sub-streams are merged, the inter-packet time is the same as at the source. This hypothesis will be used in Chapter 5. Its validity depends on the relative order of magnitude of the Markov parameters with respect to the arc delays.

3. **Model with delays.** In a more accurate model, we could augment the network model of Section 4.1.1 with a characterization of the transmission delay on each arc. However we do not address such models in this thesis.

Note that Nguyen and Zakhor do not discuss delay issues in their work. This is indeed irrelevant since they deal with networks consisting of two vertices linked by a set of parallel arcs. The synchronization issue is relevant only when the merged stream continues to travel through arcs with losses. Restating this, in case of parallel arcs, the delays do not influence the pattern of packet losses but only the time at which the receiver can process all the packets successfully transmitted. Hence, the arc delays do not influence the robustness of FEC.

4.1.4 Computing FEC Robustness when Routing on Parallel Arcs

As in [48], we now focus on networks consisting of two nodes linked by parallel arcs to numerically show that multipath routing can decrease the probability of an irrecoverable loss when a (N, K)-FEC block code is used to protect data packets. We first explain how to compute the probability on a single arc.

Probability of Irrecoverable Loss on one Arc

We assume here that losses on the arc considered are characterized by a (μ_0, μ_1)-Markov process M as discussed in Section 4.1.2 and that packets are sent through the arc according to a renewal process. Let $\{T_n,\, n \in \mathbb{Z}\}$ be the consecutive epochs at which packets are sent. By looking at the arc Markov process M only at those epochs, we obtain a 2-state Markov chain $\{M_n,\, n \in \mathbb{Z}\}$, defined as $M_n = M(T_n)$ for each $n \in \mathbb{Z}$. Restating this, M_n tells us in which state the arc is when the n^{th} packet is sent. Actually we already considered such a Markov chain in Section 4.1.2 (see figure 4.3). As above, we denote the transition probability from state i to state j by p_{ij}. We now discuss how to compute these probabilities depending on the way packets are sent :

1. Deterministic arrivals with inter-arrival time τ (or arrival rate $\lambda = \frac{1}{\tau}$). It can be shown that the transition probabilities are given by :

$$p_{11}^D(\tau) = \pi_1 + \pi_0\, e^{-(\mu_0+\mu_1)\tau} \tag{4.6}$$

$$p_{10}^D(\tau) = 1 - p_{11}^D(\tau) = \pi_0 \left(1 - e^{-(\mu_0+\mu_1)\tau}\right) \tag{4.7}$$

$$p_{00}^D(\tau) = \pi_0 + \pi_1\, e^{-(\mu_0+\mu_1)\tau} \tag{4.8}$$

$$p_{01}^D(\tau) = 1 - p_{00}^D(\tau) = \pi_1 \left(1 - e^{-(\mu_0+\mu_1)\tau}\right) \tag{4.9}$$

To derive equations (4.6-4.9), reconsider the second interpretation (uniformization) of the transition densities (μ_0, μ_1) given in Section 4.1.2. Suppose now that, at a certain time t, $M(t) = i$ and consider arrivals in the interval $(t, t+\tau]$ for the Poisson process driving the Markov process. If there is at least one arrival, then $M(t+\tau) = j$ with probability π_j. Otherwise, the Markov process stays in the same state and $M(t+\tau) = i$ with probability 1. By noticing that the probability that no arrival occurs in $(t, t+\tau]$ is given by $e^{-(\mu_0+\mu_1)\tau}$, one can easily derive equations (4.6-4.9).

2. General renewal process. Let $f(x)$ denote the probability density function (PDF) of the process inter-arrival time. Here the arrival rate λ is defined as the inverse of the expected inter-arrival times. Thus λ must satisfies
$$\frac{1}{\lambda} = \int_0^{+\infty} x f(x) dx.$$
By conditioning on that inter-arrival time, we can derive the general transition probabilities using equations (4.6-4.9) :
$$p_{ij}^G = \int_0^{+\infty} p_{ij}^D(x) f(x) dx \tag{4.10}$$
Note that we used the fact that the packet renewal process and the Markov process are independent.

3. Poisson process with arrival rate λ. In this particular case, the PDF of inter-arrival time is $f_\lambda(x) = \lambda e^{-\lambda x}$. Equation (4.10) yields :
$$p_{11}^M(\lambda) = \frac{\mu_0 + \lambda}{\mu_0 + \mu_1 + \lambda} \tag{4.11}$$
$$p_{10}^M(\lambda) = 1 - p_{11}^M(\lambda) = \frac{\mu_1}{\mu_0 + \mu_1 + \lambda} \tag{4.12}$$
$$p_{00}^M(\lambda) = \frac{\mu_1 + \lambda}{\mu_0 + \mu_1 + \lambda} \tag{4.13}$$
$$p_{01}^M(\lambda) = 1 - p_{00}^M(\lambda) = \frac{\mu_0}{\mu_0 + \mu_1 + \lambda} \tag{4.14}$$

We used superscript D, G and M following the usual Queuing Theory notation.

Consider now packets labeled from 1 to N as being the N consecutive packets of a FEC block. As in the introduction of this chapter, let X

be the number of packets successfully transmitted among packets of the block. Hence $X = \sum_{n=1}^{N} M_n$. The probability distribution of X is given as follows.

Proposition 4.1. *Let $\{M_n,\, n \in \mathbb{Z}\}$ be a stationary 2-state Markov chain on $\{0,1\}$ with transition probabilities $p_{ij}, i,j \in \{0,1\}$ and let $X = \sum_{n=1}^{N} M_n$. Then, for $1 \leq x \leq N-1$,*

$$\mathsf{P}[X=x] = \frac{1}{p_{01}+p_{10}} \sum_{k=1}^{\min\{x,N-x\}} p_{01}^{k} p_{10}^{k} p_{00}^{N-x-k-1} p_{11}^{x-k-1} \tag{4.15}$$
$$\begin{pmatrix} N-x-1 \\ k-1 \end{pmatrix} \begin{pmatrix} x-1 \\ k-1 \end{pmatrix} \left(p_{01}p_{00}\frac{x-1}{k} + p_{10}p_{11}\frac{N-x-k}{k} + 2p_{11}p_{00} \right)$$

and

$$\mathsf{P}[X=0] = \pi_0 p_{00}^{N-1}, \tag{4.16}$$
$$\mathsf{P}[X=N] = \pi_1 p_{11}^{N-1}. \tag{4.17}$$

Proof. Equation 4.16 follows from the fact that all packets are lost if and only if, at the first packet arrival, the chain is in state 0 and it stays in state 0 for the next $N-1$ packet arrivals. A similar observation proves equation 4.17.

For the case $1 \leq x \leq N-1$, let B_x be the set of vectors $\mathbf{m} \in \{0,1\}^N$ that satisfy

$$\begin{aligned} m_1 &= 0, \\ m_N &= 0, \\ \sum_{n=1}^{N} m_n &= x, \end{aligned}$$

and such that the number of transitions from state 0 to state 1 in the sequence $(m_n)_{1\leq n\leq N}$ equals k, for some positive integer k. Consider a given $\mathbf{m} \in B_x$. Since k is also the number of transitions from state 1 to 0 in $\mathbf{m}$, the probability that the random sequence $(M_n)_{1\leq n\leq N}$ equals $(m_n)_{1\leq n\leq N}$ is given by :

$$\pi_0 p_{00}^{N-x-k} p_{01}^{k} p_{11}^{x-k-1} p_{10}^{k}, \tag{4.18}$$

no matter where state transitions occur within $\mathbf{m}$.

For $i = 0, 1$, we define a *$i-$burst* as a maximal subsequence in $\mathbf{m}$ of consecutive entries with value i. Now observe that a sequence $\mathbf{m} \in B_x$

is uniquely determined if one chooses the sizes of $k+1$ 0-bursts and k 1-bursts provided that the sum of the 0-burst lengths is $N-x$ and the sum of the 1-burst lengths is x. Restating this, there is a 1-to-1 correspondence between B_x and the cross-product of the sets of ordered partitions of $N-x$ and x in $k+1$ and k terms respectively. Therefore the number of sequences in B_x is given by

$$|B_x| = \left(\begin{array}{c} N-x-1 \\ k \end{array} \right) \left(\begin{array}{c} x-1 \\ k-1 \end{array} \right) \tag{4.19}$$

In summary we have proved that

$$\mathsf{P}\left[\sum_{n=1}^{N} M_n = x, M_1 = 0, M_N = 0\right] \tag{4.20}$$

$$= \left(\begin{array}{c} N-x-1 \\ k \end{array} \right) \left(\begin{array}{c} x-1 \\ k-1 \end{array} \right) \pi_0 p_{00}^{N-x-k} p_{01}^{k} p_{11}^{x-k-1} p_{10}^{k}. \tag{4.21}$$

Similar equalities can be obtained in the three other cases for the values of M_1 and M_n. Their sum and equations (4.2-4.3) yield equation (4.15).

□

From equations (4.15-4.17), we can therefore compute the cumulative distribution of X which, as written in equation (4.1), gives us the probability of irrecoverable loss for a $(N,K)-$FEC code block :

$$P_{IL}(R,K) = \sum_{x=0}^{K-1} \mathsf{P}[X=x]. \tag{4.22}$$

However an alternative counting argument directly leads to another expression for the cumulative distribution of the number of received packets of a block.

Proposition 4.2.

$$\begin{aligned} & P_{IL}(R,K) \\ & = \sum_{k=0}^{R-1} \left[\pi_0 \left(\begin{array}{c} R-1 \\ k \end{array} \right) p_{01}^{k} p_{00}^{R-1-k} \left(\begin{array}{c} K-1 \\ k \end{array} \right) p_{10}^{k} p_{11}^{K-1-k} \right. \\ & \qquad \left. + \sum_{l=k+1}^{K-1} \left(\begin{array}{c} R-1 \\ k \end{array} \right) p_{01}^{k} p_{00}^{R-1-k} \left(\begin{array}{c} K-1 \\ l \end{array} \right) p_{10}^{l} p_{11}^{K-1-l} \right]. \end{aligned} \tag{4.23}$$

Proof. We first point out that the proof given here was actually derived by analogy with the proof of the yet-to-come Proposition 5.1.

For $i = 0, 1$, the Markov chain M_n stays in state i for a time geometrically distributed with mean $\frac{1}{p_{i(1-i)}}$. We naturally assume that a unit of time is defined as the time between transitions within the chain. Thus we can view the time intervals spent in each state as inter-arrival times of two mutually independent arrival processes, say S_0 and S_1, with inter-arrival times geometrically distributed with mean $\frac{1}{p_{01}}$ and $\frac{1}{p_{10}}$ respectively. Alternatively, for $i = 0, 1$, we can describe S_i by saying that, at each transition time, an arrival independently occurs with probability $p_{i(1-i)}$ (S_0 and S_1 may be seen as discrete-time analogs of a Poisson process). Let C be a positive integer. For $i = 0, 1$, let $N_i(C)$ be the number of arrivals for S_i among C consecutive transition times. Hence $N_i(C)$ follows a Binomial distribution given by :

$$\mathsf{P}[N_i(C) = k] \quad = \quad \left(\begin{array}{c} C \\ k \end{array} \right) p_{i(1-i)}^k p_{ii}^{C-k},$$

for any integer k satisfying $0 \leq k \leq C$. Now observe that, if $M_1 = 0$, then $X < K$ if and only if $N_0(R-1) \leq N_1(K-1)$ by looking at the transition times following state M_1. This shows that

$$\begin{aligned} \mathsf{P}[X < K | M_1 = 0] &= \sum_{k \geq 0} \sum_{l \geq k} \mathsf{P}[N_0(R-1) = k] \mathsf{P}[N_1(K-1) = l] \qquad (4.24) \\ &= \sum_{k \geq 0} \sum_{l \geq k} \left(\begin{array}{c} R-1 \\ k \end{array} \right) p_{01}^k p_{00}^{R-1-k} \left(\begin{array}{c} K-1 \\ l \end{array} \right) p_{10}^l p_{11}^{K-1-l}. \end{aligned}$$

Similarly, if $M_1 = 1$, then $X < K$ if and only if $N_0(R-1) < N_1(K-1)$. Thus one has

$$\begin{aligned} \mathsf{P}[X < K | M_1 = 1] &= \sum_{k \geq 0} \sum_{l > k} \mathsf{P}[N_0(R-1) = k] \mathsf{P}[N_1(K-1) = l] \\ &= \sum_{k \geq 0} \sum_{l > k} \left(\begin{array}{c} R-1 \\ k \end{array} \right) p_{01}^k p_{00}^{R-1-k} \left(\begin{array}{c} K-1 \\ l \end{array} \right) p_{10}^l p_{11}^{K-1-l}, \end{aligned}$$

which, with equation 4.24, immediately yields equation 4.23. □

The above results now allow us to compute the PIL on one link given FEC parameters (N, K), a (μ_0, μ_1)-Markov process and a characterization of the incoming packet stream. In figure 4.4, P_{IL} is plotted against the

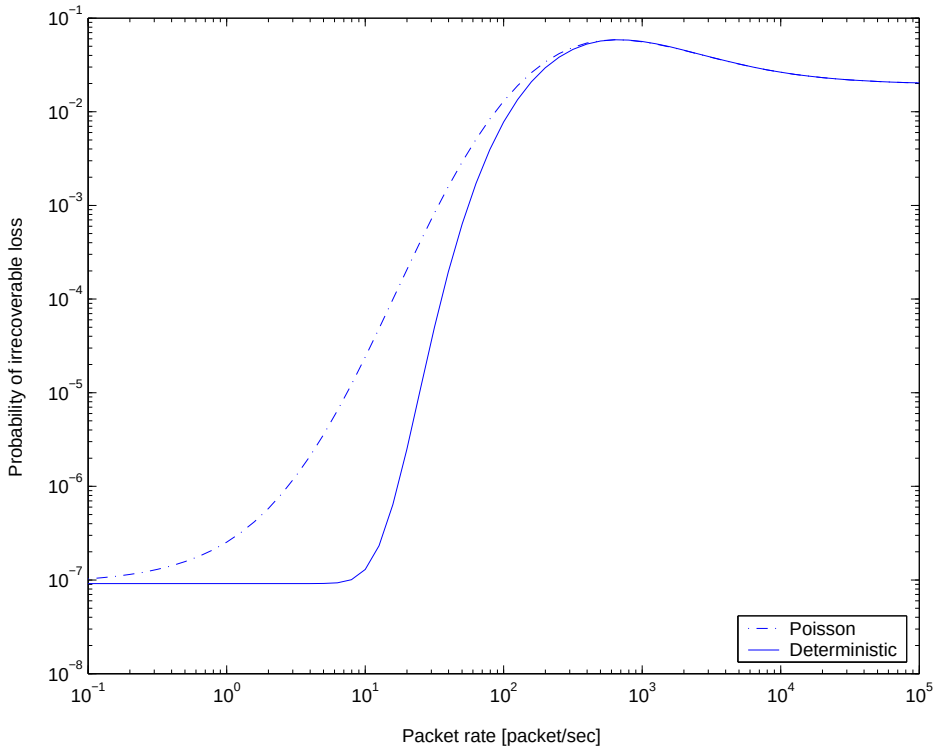

Figure 4.4: *Probability of irrecoverable loss on one path for Poisson and deterministic packet arrival processes.* $(N, K) = (100, 88)$*,* $\mu_0 = 50$ *and* $\mu_1 = 1$.

arrival rate for deterministic and Poisson packet arrivals, with parameters $(N, K) = (100, 88)$, $\mu_0 = 50$ and $\mu_1 = 1$. Hence the transition probabilities of the Markov chain are computed with equations (4.6-4.9) and (4.11-4.14) respectively. Remark that when λ tends to ∞, it is very likely that the Markov process will remain in the same state while the N packets are sent and thus either all of them or none of them will be lost. Hence P_{IL} tends to $\pi_0 = \frac{1}{51}$. Similarly, when λ tends to 0, inter-arrival times increase and we can consider that the states of the Markov process at arrival epochs are mutually independent. Therefore the distribution of the number of received packets tends to the Binomial(N, π_1) distribution and the FEC code is very efficient.

As one can see in figure 4.4, P_{IL} is not a monotonic function of λ. In our example, P_{IL} is indeed maximized for a value of λ around 6.10^2 on both curves. This phenomenon can be interpreted as a kind of "resonance" effect : for a given average loss rate π_0, the "efficiency" of error bursts to produce irrecoverable loss would be maximized if each error burst could remove exactly $R + 1$ packets within each FEC block. Since $\frac{R}{\lambda}$ is the expected amount of time needed to send $R + 1$ consecutive packets and $\frac{1}{\mu_0}$ is the expected time length of a burst in state 0, resonance should happen for $\lambda \approx R\mu_0$, i.e. for $\lambda \approx 12\,.\,50 = 600$ in our example, as we observed in figure 4.4[1].

Probability of irrecoverable loss on two arcs

Here we illustrate the fact that splitting the transmission of a block on several paths may increase FEC robustness. For this purpose, we consider two vertices linked by two parallel arcs and assume that the two corresponding Markov processes are characterized by the same pair of parameters (μ_0, μ_1). Moreover we focus on the deterministic packet arrival case with rate λ and assume that we send every other packet on each link, that is the packet stream is evenly split onto both arcs in a deterministic round-robin way. Hence, on each arc, the state of the Markov process at consecutive packet arrival epochs is evolving according to a Markov chain whose transition probabilities are given by equations (4.6-4.9) with inter-packet arrival time $\tau = \frac{2}{\lambda}$. Let X_1 (resp. X_2) be the number of packets out of $\lceil \frac{N}{2} \rceil$ (resp. $\lfloor \frac{N}{2} \rfloor$) successfully transmitted on the first arc (resp. the second arc). Their distribution can be computed using Proposition 4.1. Since losses on each arc are assumed to occur independently, the proba-

[1] Thanks to F. Glineur for this nice interpretation.

bility distribution of the total number of packets successfully transmitted, $X = X_1 + X_2$, is obtained by the following convolution.

$$\mathsf{P}[X = x] \quad = \quad \sum_{y=0}^{x} \mathsf{P}[X_1 = y]\mathsf{P}[X_2 = x - y].$$

Using equation (4.22), we can then compute the probability of irrecoverable loss of a (N, K)-FEC block code for this 2-arc case. Figure 4.5 compares the probability of irrecoverable loss for one and two arcs, with parameters $(N, K) = (100, 88)$, $\mu_0 = 50$ and $\mu_1 = 1$. Observe that, when λ tends to 0, each packet is independently lost with probability π_0 and the probabilities of irrecoverable loss tends to the same value in both cases. When λ tends to ∞, we can recover the block data only if both Markov processes stays in state 1 while the N packets are sent (assuming that $K \geq \frac{N}{2}$). Therefore the probability of irrecoverable loss on two arcs tends to $1 - \pi_1^2 \leq 1 - \pi_1 = \pi_0$, showing that it is safer to use one link. However for a large range of sending rates, it appears that, for the same loss characteristics, routing on two arcs is better than on a single arc. A closer look at figure 4.5 indicates that, in that range, a good approximation of the 2-arc curve is obtained by a $\log(2)$-shift of the single arc curve to the right. This is essentially due to the fact that, in the two arc case, the packet inter-arrival time on a given arc is twice as much as the packet inter-arrival time in the single arc case.

This example thus confirms the intuition we gave above to motivate our study of the benefits of multipath routing on FEC robustness : if traffic is split on two arcs, an error burst on one of the arcs affects only a half of the packets it would affect if all the traffic was sent on that arc. The benefits of multipath routing can thus be *observed* within the total received packet stream by noticing that packet error bursts are shorter. In the next section, we present a traffic descriptor called *peakedness* that can be interpreted as a measure of the stream *burstiness*.

Moreover notice that there is very little hope to obtain an exact formula to compute the probability of irrecoverable loss when routing on a network significantly larger than the one with two parallel arcs. On the one hand, the formulas we derived so far do not seem very useful for further analytic development. On the other hand, since the network can be seen as a single Markov process with a number of states exponential in the network size, we do not see how we could generalize the previous exact approach to general networks since dealing with exact PIL computations would be too time-consuming. Our goal is thus to propose an approxima-

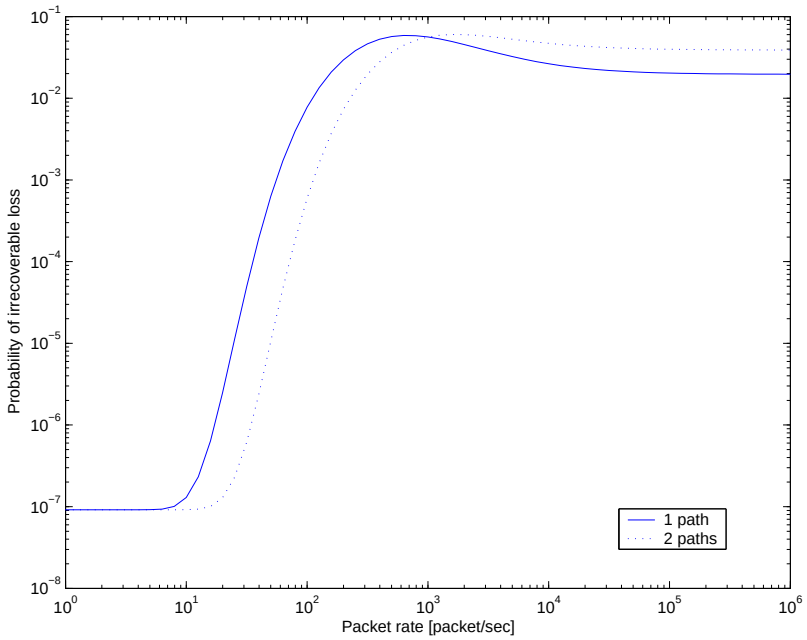

Figure 4.5: *Probability of irrecoverable loss for one and two arcs vs packet arrival rate* λ. $(N, K) = (100, 88)$, $\mu_0 = 50$, *and* $\mu_1 = 1$.

tion method based on a characterization of packet flows which takes into account , not only their average flow rate, but also their peakedness. The idea is then to give an approximate value of the PIL based on the average loss rate that the packet flow has incurred and on the way that packet losses are spread over time.

Coding Time and Rate

Before presenting the concept of peakedness, we would like to mention that, in our computational experiments, we have also observed that if we change the size N of the FEC block but keep the same redundancy ratio within a block, the P_{IL} functions remain essentially the same. For a fixed redundancy ratio $\frac{R}{N}$, it turns out that, for large enough values of N, the key parameter is actually the *expected coding time* $T = \frac{N}{\lambda}$, i.e. the expected time spent to send a $N-$packet block code, or equivalently, the *coding rate* $\lambda_C = \frac{1}{T}$. In figure 4.6, we plotted P_{IL} on one arc with parameters $(\mu_0, \mu_1) = (50, 1)$ as a function of the coding rate λ_C for three different pairs of FEC parameters (N, K) with the same redundancy ratio $\frac{R}{N} = 0.12$: (50,44), (100,88) and (200,176).

This observation motivates our presentation in Chapter 5 of a *fluid* traffic formulation of the problem, which allow us to somehow get rid of the discrete character of the FEC coding and concentrate on the key parameter of the error protection mechanism : the coding time and the redundancy ratio. Indeed, for large values of N (e.g. N ¿ 100) which are typically used in FEC encoder, figure 4.6 shows that the curves mainly differ on a range of values where the PIL is extremely small. However, in the range of more realistic values for PIL, the curves are almost identical. Moreover we will see in Chapter 5 that a fluid model allows us to propose an elegant and more general model for our problem.

4.2 Peakedness Descriptor

Since the early work of Erlang, the teletraffic literature has primarily dealt with a stationary point process representation of traffic flows. The challenge in most teletraffic studies consists in finding sufficiently accurate approximations or representations of the flows in order to evaluate their impact on system performance. In particular, it turns out that a key characteristic of traffic flows is their "burstiness", which can be loosely defined as a measure of how the point process arrival times $\{T_n\}$ spread over time.

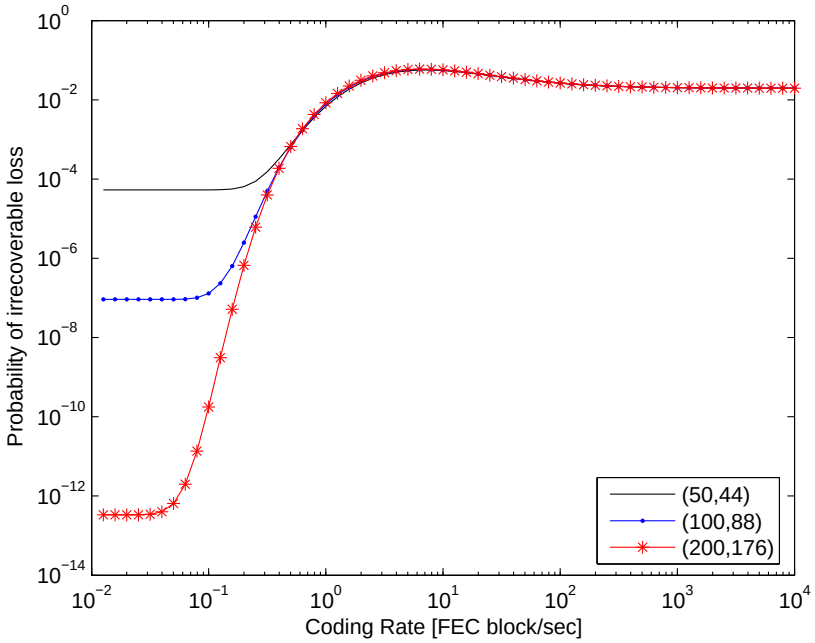

Figure 4.6: *Probability of irrecoverable loss for one arc* ($\mu_0 = 50$, *and* $\mu_1 = 1$) *vs coding rate* λ_C. *FEC parameters* $(N, K) = (50, 44)$, $(100, 88)$ *and* $(200, 176)$

The first uses of such a measure appeared for the computation of grade of service (or blocking probability) in telephone systems. In this context, at each telephone central office, a stream of call arrivals is directed to a trunk group and, in case of saturation, the call overflow stream is re-directed to a secondary trunk group. However, because of the bursty nature of this overflow, one cannot compute the blocking probability for the secondary trunk group in the same way as for the first. In [61], Wilkinson introduced the concept of *peakedness* to characterize the burstiness of call overflow streams and their impact on blocking probabilities. As shown in the next subsection, peakedness is a traffic burstiness descriptor which characterizes a traffic flow by its effect on a fictitious iid infinite-server-group.

4.2.1 Peakedness Definition and Properties

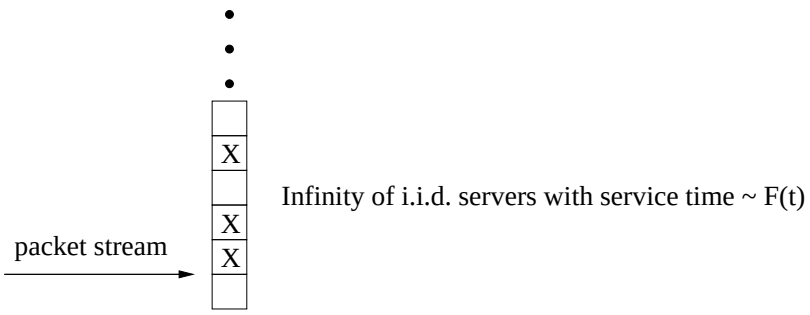

Figure 4.7: *Fictitious infinite-server system processing the packet stream*

Our introduction to the peakedness notion roughly follows [36]. Let X denote a stationary point process with rate λ_X and covariance density $k_X(t)$. As depicted in figure 4.7, we consider an infinite number of iid servers with service time distribution $F(x)$ of mean $\frac{1}{\mu_F}$ or, equivalently, with average service rate μ_F. At each arrival in X, one of the servers becomes busy according to the service distribution. Let $S(t)$ be the number of busy servers at time t. The *peakedness functional*, $z_X[F]$, maps the service time distribution, $F(x)$, to the following scalar

$$z_X[F] = \frac{\mathsf{Var}\,[S(t)]}{\mathsf{E}\,[S(t)]}. \tag{4.25}$$

Note that $z_X[F]$ is well defined since $\mathsf{Var}\,[S(t)]$ and $\mathsf{E}\,[S(t)]$ are independent of t because of the stationarity of X.

Let F^c denote the complementary service time distribution, that is $F^c(t) = 1 - F(t)$. We define its *autocorrelation function*, $\rho_{F^c}(t)$ as

$$\rho_{F^c}(t) = \int_{\max(0,-t)}^{+\infty} F^c(\tau) F^c(\tau + t) d\tau. \tag{4.26}$$

We can now compute the peakedness functional of a stationary point process from its second-order characterization.

Proposition 4.3.

$$z_X[F] = 1 + \frac{\mu_F}{\lambda_X} \int_{-\infty}^{+\infty} \left(k_X(t) - \lambda_X \delta(t)\right) \rho_{F^c}(t) dt, \tag{4.27}$$

where $\delta(t)$ is the Dirac delta function.

Proof. See [20].

□

In the teletraffic literature, the peakedness definition is often limited to an exponential distribution of the server service time. It is indeed technically convenient to see the peakedness as a function of the service time rate in that case. From now on, the peakedness with respect to exponential service time with service rate $\mu_F = s$ (or equivalently with expected service time $\frac{1}{s}$) is denoted by $z(s)$. In this case, the complementary service time distribution is given by $F^c(t) = e^{-st}$ and its autocorrelation function by $\rho_{F^c} = \frac{1}{2s} e^{-st}$.

Example 4.2. We can now compute the peakedness of the simple point processes introduced in example 4.1 :

1. **Poisson process of rate λ.** In this case, the peakedness is trivially 1. This can actually be readily derived from the basic definition of peakedness in equation (4.25).

2. **Deterministic point process of rate $\lambda = \frac{1}{\tau}$.** The peakedness is then given by

   $$z(s) = 1 - \frac{1}{s\,\tau} + \frac{e^{-s\,\tau}}{1 - e^{-s\,\tau}}.$$

   It is easy to verify that $z(s)$ is less than 1, which confirms the intuition that a deterministic arrival process is less "peaked" than a Poisson process.

4.2.2 Using Peakedness to Approximate the robustness of FEC

As motivated in Section 4.1.4, we show here how peakedness can be used to characterize the impact of losses on a packet stream in order to approximate the probability of irrecoverable loss of a FEC block. For this purpose, we focus on the parallel arc case, as in Nguyen and Zakhor's work. We assume in particular that

1. the incoming flow is a Poisson process of rate λ and hence of peakedness 1,
2. the network consists of a set of k parallel arcs labeled $a = 1, \ldots, k$. Losses on arc a are characterized by a $(\mu_0^{(a)}, \mu_1^{(a)})$-Markov process.

Peakedness Transformation

The first step is to show how the peakedness of this stream is transformed as it is routed on the arcs (see [21]) :

1. If the stream is randomly split onto the arcs according to probabilities $p^{(1)}, p^{(2)}, \ldots, p^{(k)}$, the stream on arc a is a Poisson process of rate $\lambda p^{(a)}$.
2. If a Poisson process is sent across a path with losses characterized by a (μ_0, μ_1)-Markov process, the resulting stream has a rate $\lambda\pi_1$ and peakedness $1 + \frac{\lambda\pi_0}{s+\mu_0+\mu_1}$.
3. If several independent streams of rate $\lambda^{(a)}$ and peakedness $z^{(a)}(s)$ ($a = 1, \ldots, k$) are superposed, the resulting stream has a rate $\lambda' = \sum_{a=1}^{k} \lambda^{(a)}$ and peakedness $z(s) = \frac{1}{\lambda'} \sum_{a=1}^{k} \lambda^{(a)} z^{(a)}(s)$.

Note that we will prove extensions of these results in Proposition 5.5.

Approximation Method

For a single arc with losses characterized by a (μ_0, μ_1)-Markov process, the incoming flow (of rate λ and peakedness 1) is transformed in an outgoing flow with rate λ_{out} and peakedness $z_{out}(s)$ given by :

$$\lambda_{out} = \lambda\pi_1, \tag{4.28}$$

$$z_{out}(s) = 1 + \frac{\lambda\pi_0}{s + \mu_0 + \mu_1}. \tag{4.29}$$

By inverting relations (4.28-4.29), we can write μ_0 and μ_1 as a function of λ, λ_{out} and $z_{out}(0)$:

$$\mu_0 = \frac{\lambda_{out}}{z_{out}(0)-1}\left(1-\frac{\lambda_{out}}{\lambda}\right), \tag{4.30}$$

$$\mu_1 = \frac{\lambda}{z_{out}(0)-1}\left(1-\frac{\lambda_{out}}{\lambda}\right)^2. \tag{4.31}$$

Now consider a network instance consisting of several parallel arcs. Given any splitting of a Poisson process onto the arcs, the three transformation properties of peakedness mentioned above allow us to compute the rate λ_{out} and peakedness $z_{out}(s)$ of the outgoing stream. With equations (4.30-4.31), we are able to compute (μ_0, μ_1) parameters of an "equivalent" single link which yields an outgoing stream with the same rate λ_{out} and peakedness evaluated in 0, $z_{out}(0)$. Therefore we naturally propose to approximate the probability of irrecoverable loss for the given splitting by the probability on the equivalent single link, which we can compute exactly with equations (4.15)-(4.17).

An example

Figure 4.8 shows the probability of irrecoverable loss as a function of the splitting ratio on the first arc when the Markov parameters are $(\mu_0, \mu_1) = (200, 2)$ on the first path and $(\mu_0, \mu_1) = (100, 1)$ on the second path. FEC parameters are $(N, K) = (100, 88)$ and the packet rate is 300 packets/s. The dotted line represents the probability approximation given by our method. Observe that the two curves are quite close to each other. More interestingly they have the same minimum, which means that our approximation schemes is the most efficient around the optimal point.

4.3 Conclusion

By using a Markovian loss model, we have shown that that multipath routing can increase FEC robustness for a certain range of packet sending rates. We have proposed a method to approximate the probability of irrecoverable loss by characterizing the impact of the network and the routing on the peakedness of the output stream. While we do not know how to exactly compute the PIL for networks larger than a few parallel arcs, the peakedness approximation scales well with the network size a priori.

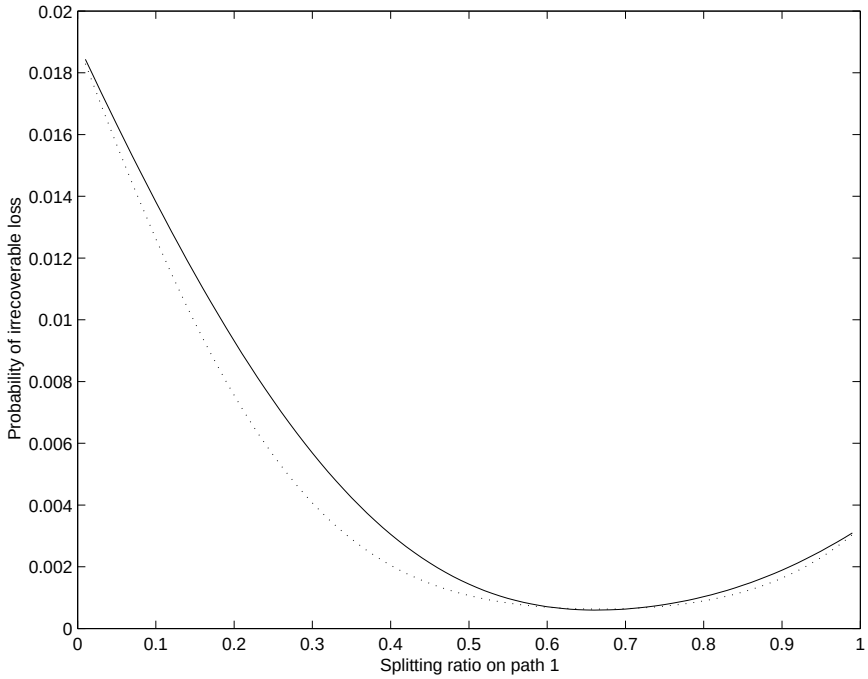

Figure 4.8: *Probability of irrecoverable loss for a routing on two paths vs splitting ratio on path 1.* $(N, K) = (100, 88)$*, packet rate* $\lambda = 300$*,* $\mu_0 = 200$ *and* $\mu_1 = 2$ *on path 1, and* $\mu_0 = 100$ *and* $\mu_1 = 1$ *on path 2. The dotted line is the probability approximation given by our method.*

Moreover we have discussed several modeling issues, which still need to be fixed for general network instances. In particular, issues concerning the transmission delay have been ignored here because of the simple network topologies addressed.

Following a remark made in Section 4.1.4, we propose in the next chapter to see the problem from a fluid formulation perspective. This allows us to propose a model which addresses all the modeling issues discussed above. Our peakedness approximation method is reformulated in this context and we give a more involved discussion of its ability to efficiently characterize the impact of losses on the streams and, hence, on FEC robustness.

In an attempt to clarify what the contribution of the next chapter is with respect to the present chapter, we give below a table which compares the model assumptions and the results obtained between the two chapters and the work of Nguyen and Zakhor. Bold entries represent original contributions of this thesis. Pointers to the sections concerned are given between parentheses.

<table>
<tr><th></th><th>N & Z</th><th>Chapter 4</th><th>Chapter 5</th></tr>
<tr><td>Assumptions</td><td colspan="3"></td></tr>
<tr><td rowspan="2">FEC</td><td colspan="3">constant parameters</td></tr>
<tr><td colspan="2">discrete interpretation</td><td>fluid interpretation (5.2.2)</td></tr>
<tr><td>Network Topologies</td><td colspan="2">2 vertices
parallel arcs</td><td>general topologies (5.2.1)</td></tr>
<tr><td>Loss model on arcs</td><td colspan="3">pairwise independent, stationary 2-state Markov processes with parameters independent of traffic
(4.1.3) (5.2.1)</td></tr>
<tr><td>Arc delays</td><td colspan="2">irrelevant
(4.1.4)</td><td>zero delay
(5.2.1)</td></tr>
<tr><td rowspan="2">Input traffic</td><td colspan="2">deterministic</td><td rowspan="2">fluid process
(5.1.2)</td></tr>
<tr><td></td><td>Poisson process
(4.1.4)</td></tr>
<tr><td>Results</td><td colspan="3"></td></tr>
<tr><td>PIL on one arc</td><td>recursion formula</td><td>explicit formula
(4.1.4)</td><td>explicit formula
(5.2.3)</td></tr>
<tr><td>PIL for several arcs</td><td>exact convolution formula</td><td colspan="2">peakedness approximation
(4.2.3) (5.5)</td></tr>
<tr><td>Evaluation of the method</td><td>-</td><td>comparison with exact formula
(4.2.3)</td><td>comparison with simulations
(5.5)</td></tr>
<tr><td rowspan="3">Tools for computing peakedness</td><td rowspan="3">-</td><td>classical definition
(4.2.1)</td><td>new definition of peakedness
(5.4)
and variability
(5.4.3)</td></tr>
<tr><td>classical properties
(4.2.3)</td><td>extension of classical properties
(5.4.1) (5.4.3)</td></tr>
<tr><td colspan="2">brief comparison
(5.4.2)</td></tr>
</table>

Chapter 5

Fluid Traffic Formulation

In Section 4.1.4, we briefly showed how the probability of irrecoverable loss evolves when one considers different traffic packetization levels while maintaining the same redundancy ratio and coding frequency (see figure 4.6). Conceptually, if we push the subdivision of traffic into smaller packets to the limit, the traffic may be considered as "fluid". In particular, the traffic may be characterized by its instantaneous arrival "rate" at any point in time. It turns out that fluid traffic models allow us to capture the most salient characteristics of the problem while being computationally more tractable than their point process counterparts.

We begin this chapter by giving a short presentation of the notion of rate process in Section 5.1. This notion somehow unifies point processes and fluid processes in a single definition. In Section 5.2, we pose our network model and we state our FEC robustness problem in this new framework. In particular we derive an expression for the probability of irrecoverable loss in the single arc case. In Section 5.3, we present the concept of second-order characterization of a rate process and show how it is transformed as the various flow processes are routed within the network. These results are then applied in Section 5.4 where we present a notion of peakedness extended to rate processes. Then, in Section 5.5, we show how the peakedness characterization can be used to approximate the probability of irrecoverable loss of a FEC block. Computational results are given. Section **??** discusses the use of the peakedness characterization to minimize the PIL. The chapter ends with a short conclusion in Section 5.6.

5.1 Rate Processes

As far as we know, the "rate process" terminology is due to [44], which we follow in this introductory section.

A *rate process* $\{X(t); t \in \mathbb{R}\}$ is formally defined as a stationary generalized random process. Understanding the exact meaning of such a definition actually requires, as a background, an introduction to the notion of *generalized random process*, which is beyond the scope of this thesis (see e.g. [29] as reference). However we can informally see a generalized random process as a random process whose sample paths are generalized functions over $\mathbb{R}$. In practice, this definition allows us to consider random processes containing Dirac delta functions at some points in $\mathbb{R}$.

A some time t, the rate value $X(t)$ may be interpreted as saying that, in the infinitesimal time interval $[t, t+\delta)$, the amount of traffic that arrives is $X(t)dt$. The rate process X is then associated to a corresponding *counting process* $N_X(t)$ defined as

$$N_X(t) \quad = \quad \int_0^t X(t)dt.$$

Remark that, as for point processes, we shall require stationarity in the definition of a rate process, which means that the random characterization of the process is essentially invariant to time translations. Therefore we can define the *arrival rate* of rate process X as a constant λ_X given by $\lambda_X = \mathsf{E}[X(t)]$. Similarly, the *covariance density* of the associated counting process $N_X(t)$ is $k_X(t) = \mathsf{Cov}[X(\tau), X(\tau+t)]$, where the right-hand side is indeed independent of τ.

As mentioned above, this definition of rate process includes point processes as well as fluid processes, as detailed hereafter.

5.1.1 Point Processes

Any point process can be seen as a rate process. Indeed, let $(T_i)_{i\in\mathbb{Z}}$ be the sequence of arrival times of a given point process, then this process can be viewed as a rate process $X(t)$ defined by

$$X(t) \quad = \quad \sum_{i=-\infty}^{\infty} \delta(t - T_i).$$

The corresponding counting process is

$$N(t) = \sum_{i=-\infty}^{\infty} u(t - T_i),$$

where $u(t)$ is the unit step function, sometimes called *Heaviside function*, and is defined as $u(t) = 0$, for $t < 0$, and $u(t) = 1$, for $t \geq 0$.

5.1.2 Fluid Flow Processes

Following [44], we define a *fluid flow process* as a rate process $\{X(t); t \in \mathbb{R}\}$ where $X(t)$ is a stationary random process having sample paths which are continuous everywhere except on a countable set of points. In particular, this definitions rules out the possibility of having Dirac delta functions in sample paths of a fluid flow process and thus captures the fact that, for a fluid flow, the amount of traffic received increases continuously with time.

Example 5.1. Let us consider some simple classes of fluid flows.

1. **constant fluid process.** A *constant fluid process* C satisfies $C(t) = \lambda$ for a constant $\lambda \geq 0$. Hence $\lambda_C = \lambda$, $k_C(t) = 0$ and $N_C(t) = \lambda t$.

2. **Markov Modulated Fluid Process.** A *Markov Modulated Fluid Process (MMFP)* X is characterized by an underlying Markov process $\{M(t), t \in \mathbb{R}\}$ on a finite state space E and a rate function $h : E \to \mathbb{R}_+$ describing how the states of M modulate the flow rate of X, that is $X(t) = h(M(t))$. Here we are particularly interested in a special case, sometimes called an *ON-OFF Markov fluid process*, where M is a 2-state Markov process just as for the Gilbert Model introduced in Section 4.1.2. When the process is in state 0 (or OFF), the rate is zero. Conversely, when the process is in state 1 (or ON), the rate is a constant $\lambda \geq 0$. Hence the process can be seen as a MMFP with $E = \{0, 1\}$ and $h(i) = i\lambda,\ i \in E$. If X is driven by a (μ_0, μ_1)-Markov process M, $\lambda_X = \pi_1\lambda$ and

$$\begin{aligned} k_X(t) &= \mathsf{Cov}[X(\tau), X(\tau + t)] \text{ , for any } \tau \in \mathbb{R}, \\ &= \mathsf{E}[X(\tau)X(\tau + t)] - \mathsf{E}[X(\tau)]\mathsf{E}[X(\tau + t)] \\ &= \lambda^2(\mathsf{P}[M(\tau) = 1, M(\tau + t) = 1] - \pi_1^2) \\ &= \lambda^2(\pi_1 p_{11}^D(|t|) - \pi_1^2) \\ &= \lambda^2\pi_0\pi_1 e^{-(\mu_0+\mu_1)|t|} \end{aligned}$$

where $p_{11}^D(.)$ is given in equation (4.6).

5.2 Fluid Formulation for the FEC Robustness Problem

We first pose our network model following the discussion of Section 4.1.

5.2.1 Network Model and Routing Assumptions

Our network is modeled as a directed graph $G = (V, A)$ where each arc $a \in A$ is associated with a 2-state Markov process $\{M^{(a)}(t), t \in \mathbb{R}\}$, where $M^{(a)}(t) \in \{0, 1\}$ for all $t \in \mathbb{R}$ and transition densities from state 0 to 1 and 1 to 0 are $\mu_0^{(a)}$ and $\mu_1^{(a)}$ respectively. Processes $M^{(a)}$, $a \in A$, are assumed to be stationary and mutually independent.

We consider two distinguished vertices v_{in} and v_{out} in V, respectively called the *sender* and the *receiver*, and a rate process $\{X_{in}(t), t \in \mathbb{R}\}$ which represents the input flow to be sent from v_{in} to v_{out}. Let $\mathcal{P}$ be the set of all directed paths from v_{in} to v_{out}. For each path $p \in \mathcal{P}$, we define $V(p)$ and $A(p)$ as the set of vertices and arcs of p. The routing from v_{in} to v_{out} is completely characterized if one specifies a splitting ratio $x^{(p)}$ for each path $p \in \mathcal{P}$. These splitting ratios must naturally fulfill the following constraints : $0 \leq x^{(p)} \leq 1$ for all $p \in \mathcal{P}$ and $\sum_{p \in \mathcal{P}} x^{(p)} = 1$. Now for each path $p \in \mathcal{P}$ and each vertex $v \in V(p)$, we can construct the rate process $X_v^{(p)}(t)$, which gives the instantaneous flow rate carried on path p at vertex v :

$$X_{v_{in}}^{(p)}(t) = x^{(p)} X_{in}(t) \text{ , for all } p \in \mathcal{P}, \tag{5.1}$$

$$X_w^{(p)}(t) = M^{(v,w)}(t) X_v^{(p)}(t) \text{ , for all } p \in \mathcal{P}, (v, w) \in A(p), \tag{5.2}$$

$$X_{out}(t) = \sum_{p \in \mathcal{P}} X_{v_{out}}^{(p)}(t), \tag{5.3}$$

where X_{out} is the rate process representing the output flow finally received at v_{out}. Note that this model implicitly assumes that the flow travels across network arcs without incurring delays.

Equations (5.1-5.3) respectively involve the three following types of operation on rate processes, which generalize similar operations discussed in Chapter 4 :

1. p**-thinning([35])** : the *p-thinning* of a rate process X is defined as the process $\{pX(t), t \in \mathbb{R}\}$.

2. M**-Markov modulation** : the *M-Markov modulation* of at rate process X is the rate process $\{M(t)X(t), t \in \mathbb{R}\}$, where M is an independent stationary Markov process.

3. **superposition** : the *superposition* or *sum* of rate processes $X_1, \ldots, X_n$ is naturally defined as the process $\{\sum_{i=1}^{n} X_i(t), t \in \mathbb{R}\}$.

5.2.2 Fluid Interpretation of FEC Coding

Consider a packet stream protected by a (R, K)-FEC code and let $r = \frac{R}{R+K}$ be the redundancy ratio. Suppose that packets are sent on the network according to a given renewal process with a coding expected time T, that is T is the expected time within which a whole N-packet block is sent. Moreover we choose the packet size unit in such a way that the average stream rate is 1. A fluid flow is now intuitively obtained by letting N tend to infinity and the packet size tend to 0 while maintaining T, r and the average stream rate constant. FEC blocks are now seen as a partition of the amount of input flow in continuous blocks of time length T. Since the flow is assumed to have a unit rate, each block contains T units of coded data.

Because of our stationarity hypothesis, we can assume that a certain FEC block is sent between time 0 and T. After its transmission, the receiver will be able to recover the original data encoded within that block if and only if it has received at least $(1 - r)T$ units of data during the interval $[0, T]$. Remark that this decoding condition is relevant partly because of our assumption that flows are not delayed when they reach the receiver. By analogy with the discrete case, we redefine R and K by :

$$\begin{aligned} R &= r\,T \\ K &= (1-r)\,T \end{aligned}$$

Given the complete network model G detailed in Section 5.2.1 (with splitting ratios $x^{(p)}$), the previous discussion leads us to define the *probability of irrecoverable loss for a (R, K)-FEC fluid flow sent through G* as

$$P(R,K) \quad = \quad \mathsf{P}\left[N_{out}(R+K) < K | X_{in}(t) = 1, \forall t \in [0, R+K]\,\right], \quad (5.4)$$

where $N_{out}(t) = N_{X_{out}}(t) = \int_0^t X_{out}(\tau)d\tau$.

We can therefore formulate the FEC robustness optimization as the problem of finding the splitting ratios $x^{(p)}$, $p \in \mathcal{P}$, which minimize

$P(R,K)$. In the next section, we show how to compute $P(R,K)$ for a single arc.

5.2.3 Loss Characterization on One Arc

We now compute the probability of irrecoverable loss for a unitary (R,K)-FEC fluid process X_{in} sent on a single arc with Markov parameters (μ_0,μ_1). Note that, since all the flow is sent on the same arc, the rate of the output fluid process at time t, $X_{out}(t)$, is simply the state of the arc Markov loss process at time t.

By conditioning on the initial state of the Markov process, we define

$$P_i(R,K) \;=\; \mathsf{P}\left[N_{out}(R+K) < K | X(0)=i\right],$$

for $i=0,1$.

When $X(0)=0$, if the process stays in state 0 during a time interval of length greater than R before its first transition to state 1, then $N_{out}(R+K)<K$ whatever happens to the process after this first transition to state 1. On the other hand, if this transition happens after a time $\tau \leq R$ then $N_{out}(R+K)$ will be less than K iff the process stays in state 1 for a total time less than K within the remaining time $R+K-\tau$. Since the time before the first transition from state 0 to state 1 is exponentially distributed with parameter μ_0, the previous observation can be written as :

$$P_0(R,K) \;=\; e^{-\mu_0 R} + \int_0^R \mu_0 e^{-\mu_0 \tau} P_1(R-\tau,K)d\tau. \tag{5.5}$$

Similarly, by conditioning on $X(0)=1$ and the time spent in state 1 before the first transition to state 0, we obtain :

$$P_1(R,K) \;=\; \int_0^K \mu_1 e^{-\mu_1 \tau} P_0(R,K-\tau)d\tau. \tag{5.6}$$

Substitution of equation (5.6) into equation (5.5), and vice-versa, respectively yields :

$$\begin{aligned} &P_0(R,K) \\ &= e^{-\mu_0 R} + \int_{\tau_0=0}^R \mu_0 e^{-\mu_0\tau_0} \int_{\tau_1=0}^K \mu_1 e^{-\mu_1\tau_1} P_0(R-\tau_0,K-\tau_1)d\tau_1 d\tau_0 \quad (5.7)\\ &= e^{-\mu_0 R} + \int_{\tau_0=0}^R \int_{\tau_1=0}^K \mu_0 e^{-\mu_0\tau_0}\mu_1 e^{-\mu_1\tau_1} P_0(R-\tau_0,K-\tau_1)d\tau_1 d\tau_0 \quad (5.8) \end{aligned}$$

and

$$\begin{aligned} & P_1(R,K) \\ &= \int_{\tau_1=0}^{K} \mu_1 e^{-\mu_1\tau_1}\left(e^{-\mu_0 R} + \int_{\tau_0=0}^{R} \mu_0 e^{-\mu_0\tau_0} P_1(R-\tau_0, K-\tau_1)d\tau_0\right)d\tau_1 \quad (5.9) \\ &= e^{-\mu_0 R}\left(1-e^{-\mu_1 K}\right) \quad (5.10) \\ &\qquad + \int_{\tau_0=0}^{R}\int_{\tau_1=0}^{K} \mu_0 e^{-\mu_0\tau_0}\mu_1 e^{-\mu_1\tau_1} P_1(R-\tau_0, K-\tau_1)d\tau_1 d\tau_0. \end{aligned}$$

Using the fact that $P(R,K) = \pi_0 P_0(R,K) + \pi_1 P_1(R,K)$, we deduce from equation (5.8) and (5.10) that $P(.,.)$ is a solution of the following integral equation :

$$\begin{aligned} P(x,y) &= e^{-\mu_0 x}\left(1-\pi_1 e^{-\mu_1 y}\right) \quad (5.11) \\ &\qquad + \int_{\tau_0=0}^{x}\int_{\tau_1=0}^{y} \mu_0\mu_1 e^{-(\mu_0\tau_0+\mu_1\tau_1)} P(x-\tau_0, y-\tau_1)d\tau_1 d\tau_0. \end{aligned}$$

Observe now that the last term in equation (5.11) is simply a convolution of $P(x,y)$ and function $\mu_0\mu_1 e^{-(\mu_0 x+\mu_1 y)}$. We can thus easily compute the 2-dimensional Laplace transform of P. By using a Laplace transform table ([18]), we have obtained the following result :

Proposition 5.1. *For $R, K > 0$,*

$$\begin{aligned} & P(R,K) \\ &= e^{-\mu_0 R}e^{-\mu_1 K}\left[\sum_{k\geq 0}\pi_0\left[\frac{(\mu_0 R)^k}{k!}\frac{(\mu_1 K)^k}{k!} + \sum_{l>k}\frac{(\mu_0 R)^k}{k!}\frac{(\mu_1 K)^l}{l!}\right]\right] \quad (5.12) \end{aligned}$$

However, by observing the nice form of the solution obtained, we give hereafter a direct proof of this result, which yields more probabilistic insights than the Laplace approach.

Proof. Since, for $i = 0, 1$, the Markov process stays in state i for a time exponentially distributed with parameter μ_i, we can view the time intervals spent in each state as inter-arrival times of two mutually independent Poisson processes, say S_0 and S_1,with arrival rate μ_0 and μ_1 respectively. For $i = 0, 1$, let $N_i(t)$ be the number of arrivals for S_i in the time interval [0,t]. Hence $N_i(t)$ follows a Poisson distribution of parameter $\mu_i t$, that is

$$\mathsf{P}[N_i(t) = n] = e^{-\mu_i t}\frac{(\mu_i t)^n}{n!},$$

for any non-negative integer n. Now observe that, if $X(0) = 0$, then $N_{out}(R+K) < K$ if and only if $N_0(R) \leq N_1(K)$. This shows that

$$P_0(R,K) = \sum_{k\geq 0}\sum_{l\geq k} \mathsf{P}[N_0(R)=k]\mathsf{P}[N_1(K)=l] \quad (5.13)$$

$$= e^{-\mu_0 R}e^{-\mu_1 K}\sum_{k\geq 0}\sum_{l\geq k}\frac{(\mu_0 R)^k}{k!}\frac{(\mu_1 K)^l}{l!}. \quad (5.14)$$

Similarly, if $X(0) = 1$, then $N_{out}(R+K) < K$ if and only if $N_0(R) < N_1(K)$. Thus one has

$$P_1(R,K) = \sum_{k\geq 0}\sum_{l>k} \mathsf{P}[N_0(R)=k]\mathsf{P}[N_1(K)=l]$$

$$= e^{-\mu_0 R}e^{-\mu_1 K}\sum_{k\geq 0}\sum_{l>k}\frac{(\mu_0 R)^k}{k!}\frac{(\mu_1 K)^l}{l!},$$

which, with equation 5.14, immediately yields equation 5.12. □

While formulas for $P(R,K)$ can generally be found in textbooks on stochastic processes (in [51] for instance), they are, as far as we know, given in a more complicated form than the one we have derived.

Now by considering a fixed $T = R + K$, we can replace R by $T - K$ in equation (5.12) and take the derivative of the resulting expression with respect to K. This almost gives us the following result, which is the continuous analog of proposition 4.1 :

Proposition 5.2. *Let $f_T(x)$ be the probability density function of $N_{out}(T)$, then*

$$f_T(x) = \frac{\mu_1}{\mu_0+\mu_1}e^{-\mu_0 T}\delta(x) + \frac{\mu_0}{\mu_0+\mu_1}e^{-\mu_1 T}\delta(T-x)$$
$$+\frac{\mu_0\mu_1}{\mu_0+\mu_1}e^{-\mu_0(T-x)}e^{-\mu_1 x}\sum_{k\geq 0}\frac{[\mu_0\mu_1 x(T-x)]^k}{(k!)^2}\left[2+\frac{\mu_0 x}{k+1}+\frac{\mu_1(T-x)}{k+1}\right].$$

Proof. The last term is obtained by derivation of equation (5.12) as explained above. The two first terms in the right-hand side of the equality to be proven simply represent the probability that $N_{out}(T) = 0$ and $N_{out}(T) = T$ respectively. Since $N_{out}(T) = 0$ if $X(0) = 0$ and X stays in state 0 until at least time T, $\mathsf{P}[N_{out}(T) = 0] = \pi_0 e^{-\mu_0 T}$. Similarly, $N_{out}(T) = T$ if $X(0) = 1$ and X stays in state 1 until at least time T, hence $\mathsf{P}[N_{out}(T) = T] = \pi_1 e^{-\mu_1 T}$. Note that 0 and T are naturally the only values of x in $[0,T]$ such that $\mathsf{P}[N_{out}(T) = x]$ is different from zero. □

Remark that we could also have derived the previous formula by a direct argument very similar to the one given in the proof of Proposition 4.1.

5.3 Second-order Characterization of Rate Processes

Whereas the notion of rate process is quite general and potentially allows us to model a large class of traffic flow problems, having a complete characterization of all the rate processes on all arcs quickly turns out to be impossible as the model complexity increases. This is illustrated in our problem of evaluating the probability of irrecoverable loss for FEC block code : while a complete characterization of the loss distribution is numerically available for very small networks, there is little hope to compute such a distribution for arbitrary networks and routing. Consequently, a common practice is then to back away from a complete characterization to a low order characterization of the processes.

First-order flow characterizations are typically used in the network flow optimization literature (see [1] for instance), which is mostly concerned with deterministic problems where a flow is modeled by a constant value representing its average rate. When flow losses are involved in such models, only a first-order characterization of the loss processes must be given in order to take into account their impact on flow values. In other words, the model only has to provide us with an average loss rate for each loss process. A classical example of a problem formulated in this setting is that of finding the maximum flow that can be routed from one node to another given a loss rate and a capacity on each arc of the network (see [57] for instance). Remark that, for a first-order network flow optimization problem, capacities are usually mandatory to define "interesting" problems, in the sense that all the flow would be routed on paths with minimum global loss rate in the absence of capacity constraints.

Because they do not take the dependency of loss events into account, first-order network flow models with losses are not suited to the problem of finding a routing minimizing the probability of irrecoverable loss of a FEC code block. In contrast, we describe in this chapter how second-order characterizations of rate processes allow us to compute good approximations of the PIL.

As in Section 4.1.3, a second-order characterization of a rate process X is given by the pair $(\lambda_X, k_X(t))$ since it allows us to compute any expectation or covariance for $N_X(t)$ at different times $t \in \mathbb{R}$. In our net-

work model, we naturally consider several processes simultaneously. In particular, we focus on the aggregated flow obtained as output at the receiver node. This flow may be regarded as the sum of all the flows reaching the receiver node. To compute a second-order characterization of the sum of different processes, the second-order characterization of each process is not sufficient if we do not have information relative to the covariance between processes. A second-order characterization for a set of rate processes $\mathcal{X}$ is therefore given by the average flow rate λ_X of each process $X \in \mathcal{X}$ and the *mutual covariance density function* $k_{XY}(t) = \mathsf{Cov}[X(\tau), Y(\tau + t)]$ for each $X, Y \in \mathcal{X}$. Remark that

1. Again, the mutual covariance density function is well defined because of our stationarity hypothesis.

2. Unlike the covariance density function, the mutual covariance density is not an even function in general, i.e. $k_{XY}(t) \neq -k_{XY}(-t)$. However in this thesis, all mutual covariance densities considered are even because all the rate processes included in our Markovian model are reversible.

3. The mutual covariance density of a process with itself is simply its covariance density.

We now detail how a second-order characterization is transformed with respect to the three network operations mentioned in Section 5.2.1.

Proposition 5.3. *Let X and Y be two rate processes.*

1. *Let U be the p-thinning of X, then*

$$\begin{aligned} \lambda_U &= p\lambda_X, && (5.15)\\ k_U(t) &= p^2 k_X(t), && (5.16)\\ k_{UY}(t) &= p k_{XY}(t). && (5.17) \end{aligned}$$

2. *Let M be a (μ_0, μ_1)-Markov process independent of X and Y and let V and W be the M-Markov modulation of X and Y respectively, then*

$$\begin{aligned} \lambda_V &= \pi_1 \lambda_X, && (5.18)\\ k_{VW}(t) &= k_{XY}(t)\left(\pi_1^2 + \pi_0\pi_1 e^{-(\mu_0+\mu_1)|t|}\right) && (5.19)\\ &\quad + \pi_0\pi_1 e^{-(\mu_0+\mu_1)|t|}\lambda_X\lambda_Y, \\ k_{VY}(t) &= \pi_1 k_{XY}(t). && (5.20) \end{aligned}$$

3. *Let $Z = X + Y$, then*

$$\lambda_Z = \lambda_X + \lambda_Y, \tag{5.21}$$
$$k_Z(t) = k_X(t) + k_Y(t) + k_{XY}(t) + k_{YX}(t). \tag{5.22}$$

Proof. Except for equation (5.19), all previous equations are immediate consequences of the linearity and bilinearity of the expectation and covariance operators respectively. Hence we only detail the derivation of equation (5.19). For any $\tau \in \mathbb{R}$,

$$\begin{aligned} k_{VW}(t) &= \mathsf{Cov}[V(\tau), W(\tau+t)] \\ &= \mathsf{Cov}[X(\tau)M(\tau), Y(\tau+t)M(\tau+t)] \\ &= \mathsf{Cov}[X(\tau), Y(\tau+t)]\mathsf{Cov}[M(\tau), M(\tau+t)] \\ &\quad +\mathsf{Cov}[X(\tau), Y(\tau+t)]\mathsf{E}[M(\tau)]\mathsf{E}[M(\tau+t)] \\ &\quad +\mathsf{E}[X(\tau)]E[Y(\tau+t)]\mathsf{Cov}[M(\tau), M(\tau+t)], \text{by } M \text{ independence} \\ &= k_{XY}(t)(k_M(t) + \lambda_M^2) + \lambda_X\lambda_Y k_M(t). \end{aligned}$$

Equation (5.19) follows from the results in example 5.1. □

In the next Section, we present a notion of peakedness generalized to rate processes. Peakedness can be used instead of the covariance density to provide us with an alternative second-order characterization of rate processes.

5.4 Peakedness of Rate Processes

While a notion of peakedness for fluid traffic has been proposed in [44], we present hereafter a slightly different definition. The difference between both definitions is briefly discussed below.

In comparison with the classical definition given in Section 4.2, the intuition behind our definition is that each infinitesimal amount of arriving flow is independently processed by a server characterized by a given service time distribution $F(t)$. The peakedness is then naturally defined as the variance-to-mean ratio of $S(t)$, the amount of arrived flow still being served at an arbitrary time t. However we must point out that, since each infinitesimal amount of flow is served independently, the variance of $S(t)$ is zero if the flow rate $X(\tau)$ is known for all $\tau \leq t$. In other words, all the "randomness" of $S(t)$ is now due to the arrival flow and not to the server service times. The service time distribution $F(t)$ simply plays the

role of a "leaky buffer" for the amount of flow already arrived : at time t, $F^c(t-\tau) = 1 - F(t-\tau)$ is the proportion of the amount of flow arrived at time τ which is still in the buffer at time t. Hence we have

$$S(t) \quad = \quad \int_{-\infty}^{t} X(t)F^c(t-\tau)d\tau. \tag{5.23}$$

Given a rate process $\{X(t); t \in \mathbb{R}\}$ and a cumulative distribution function $F(t)$, the process $\{S(t); t \in \mathbb{R}\}$ defined by equation (5.23) is called the *F-buffer process* of X.

We can now define the peakedness of a rate process.

Definition 5.1. *Given a stationary fluid process $\{X(t); t \in \mathbb{R}\}$ and a cumulative distribution function $F(t)$, we define the peakedness of $X(t)$ with respect to F as the value*

$$\bar{z}[F] \quad = \quad \frac{\mathsf{Var}[S(t)]}{E[S(t)]}, \tag{5.24}$$

where S is the F-buffer process of X.

Remark that, in equation 5.24, $\bar{z}[F]$ is independent of t because of the stationarity of X, as in the discrete case in Section 4.2.1.

In [44], the peakedness for a fluid process is defined by considering that all the amount of flow which arrives at a given time t is processed by the *same* server with random service time. The drawback of this definition is that the peakedness can behave in a non-continuous way when considering point process. However for pure fluid process, our definition is the same.

5.4.1 Peakedness properties

We now prove the analog of proposition 4.3 which allows us to compute the peakedness from the covariance density of the flow.

We first prove a result that makes the connection between the mutual density covariance of two rate processes and the covariance of their respective buffer processes. We recall here that, as mentioned in Section 4.2.1 ρ_{F^c} is the autocorrelation function of $F^c(t)$ and is defined by equation (4.26).

Lemma 5.1. *Let X_1 and X_2 be two stationary rate processes with mutual covariance density function $k_{12}(t)$, then the covariance of their respective $F-$buffer*

processes S_1 and S_2 is given by

$$\mathsf{Cov}[S_1(t), S_2(t)] = \int_{-\infty}^{\infty} k_{12}(t)\rho_{F^c}(t)dt. \tag{5.25}$$

Proof.

$$\begin{aligned}
&\mathsf{Cov}[S_1(t), S_2(t)] && (5.26)\\
&= \int_{-\infty}^{t}\int_{-\infty}^{t} \mathsf{Cov}[X_1(t_1), X_2(t_2)]F^c(t-t_1)F^c(t-t_2)dt_2dt_1 && (5.27)\\
&= \int_{-\infty}^{t}\int_{-\infty}^{t} k_{12}(t_1-t_2)F^c(t-t_1)F^c(t-t_2)dt_2dt_1 && (5.28)\\
&= \int_{0}^{\infty}\int_{-\tau_1}^{\infty} k_{12}(\tau)F^c(\tau_1)F^c(\tau_1+\tau)d\tau d\tau_1 && (5.29)\\
&= \int_{-\infty}^{\infty} k_{12}(\tau)\int_{\max(0,-\tau)}^{\infty} F^c(\tau_1)F^c(\tau_1+\tau)d\tau_1 d\tau && (5.30)\\
&= \int_{-\infty}^{\infty} k_{12}(\tau)\rho_{F^c}(\tau)d\tau, && (5.31)
\end{aligned}$$

with the following variable substitution in equation (5.29) : $\tau = t_1 - t_2$ and $\tau_1 = t - t_1$. □

Proposition 5.4. *Let $\{X(t); t \in \mathbb{R}\}$ be a stationary rate process with covariance density function $k_X(t)$, then the (fluid) peakedness functional of X is*

$$\bar{z}[F] = \frac{\mu_F}{\lambda_X}\int_{-\infty}^{\infty} k_X(t)\rho_{F^c}(t)dt. \tag{5.32}$$

Proof. Let S be the F-buffer process of X. At a given time t, $S(t)$ has the following expectation and covariance.

$$\begin{aligned}
\mathsf{E}[S(t)] &= E[X(t)]\int_{-\infty}^{t} F^c(t-\tau)d\tau\\
&= \lambda_X\int_{0}^{\infty} F^c(\tau)d\tau\\
&= \frac{\lambda_X}{\mu_F}.\\
\mathsf{Var}[S(t)] &= \int_{-\infty}^{\infty} k(\tau)\rho_{F^c}(\tau)d\tau,
\end{aligned}$$

using lemma 5.1.

□

Corollary 5.1. *Let $\{X(t); t \in \mathbb{R}\}$ be a stationary rate process with covariance density function $k_X(t)$, then the peakedness of X with respect to an exponentially distributed service time of mean $\frac{1}{s}$ is*

$$\bar{z}_{exp}(s) = \frac{s}{2\lambda_X} \int_{-\infty}^{\infty} k_X(t) e^{-s|t|} dt \tag{5.33}$$

$$= \frac{s}{\lambda_X} \int_{0}^{\infty} k_X(t) e^{-st} dt. \tag{5.34}$$

Proof. This follows immediately from the fact that $\rho_{F^c}(t) = \frac{1}{2s} e^{-s|t|}$ when F is the exponential distribution of mean $\frac{1}{s}$. □

Corollary 5.1 simply says that, for exponential service times, the peakedness function is essentially the unidirectional Laplace transform of the covariance density. It is worth noting that if $k(t)$ contains a Dirac delta function at the origin, then the contribution of this delta must naturally be halved in the evaluation of the unidirectional integral of equation (5.34).

We now show that there is a very simple relationship between both definitions of peakedness.

5.4.2 Link with Original Peakedness Definition

By comparing propositions 4.3 and 5.4, we immediately see that for a point process X with rate λ_X, the relationship between the original and the generalized peakedness definitions, with respect to a given service time distribution $F(t)$, is given by :

$$\bar{z}[F] = z[F] - 1 + \mu_F \rho_{F^c}(0).$$

In particular, for exponential service time, we have

$$\bar{z}_{exp}(s) = z_{exp}(s) - \frac{1}{2}.$$

These equations show that, for a given service time, both definitions only differ by a constant, i.e. a value independent of the point process X considered, and therefore provide us with the same information about the process. However the advantage of our peakedness definition is that it encompasses the characterization of an arbitrary rate process whereas the original definition is limited to point processes.

5.4.3 Peakedness Transformation Properties in Networks with Markovian Losses

We now give properties needed in order to compute the peakedness of traffic flows at different points in the network model given in Section 5.2.1. The task essentially consists in translating the results of proposition 5.3 for the peakedness instead of the covariance density. In Section 5.3, we needed to introduce the mutual covariance density. Similarly we need here a tool for computing the peakedness of a sum of potentially dependent processes. Given two rate processes X and Y we define their *mutual variability* as the functional :

$$\gamma_{XY}[F] \quad = \quad \mu_F \mathsf{Cov}[S_X(t), S_Y(y)],$$

where S_X and S_Y are the F-buffer of X and Y respectively. The *variability* of a rate process X is naturally defined as the mutual variability of X with itself and is denoted by $\gamma_X[F]$. One can immediately check that

$$\gamma_X[F] = \mu_F \mathsf{Var}[S_X(t)] = \lambda_X \bar{z}_X[F].$$

We now detail how peakedness and/or mutual variability is transformed. From now on, we focus on an exponentially distributed buffering time, that is we choose $F(t) = 1 - e^{st}$ and see the peakedness and variability as functions of s.

With these assumptions, lemma 5.1 can be rewritten as follows :

Lemma 5.2. *Let X and Y be two stationary rate processes, then*

$$\gamma_{XY}(s) \quad = \quad \frac{1}{2}\int_{-\infty}^{\infty} k_{12}(t) e^{-s|t|} dt.$$

Proof. It follows from lemma 5.1 and $\rho_{F^c}(t) = \frac{1}{2s} e^{-s|t|}$.

□

Proposition 5.5. *Let X and Y be two rate processes.*

1. *Let U be the p-thinning of X, then*

$$\bar{z}_U(s) \quad = \quad p\, \bar{z}_X(s), \tag{5.35}$$

$$\gamma_{UY}(s) \quad = \quad p\, \gamma_{XY}(s). \tag{5.36}$$

2. *Let M be a (μ_0, μ_1)-Markov process independent of X and Y and let V and W be the M-Markov modulation of X and Y respectively, then*

$$\gamma_{VW}(s) = \pi_1^2\gamma_{XY}(s) + \pi_0\pi_1\gamma_{XY}(s+\mu_0+\mu_1) + \frac{\pi_0\pi_1\lambda_X\lambda_Y}{s+\mu_0+\mu_1}, \quad (5.37)$$

$$\bar{z}_V(s) = \pi_1\bar{z}_X(s) + \pi_0\bar{z}_X(s+\mu_0+\mu_1) + \frac{\pi_0\lambda_X}{s+\mu_0+\mu_1}. \quad (5.38)$$

3. *Let $Z = X + Y$, then*

$$\gamma_Z(s) = \gamma_X(s) + \gamma_Y(s) + 2\gamma_{XY}(s), \quad (5.39)$$

$$\bar{z}_Z(s) = \frac{\lambda_X}{\lambda_X+\lambda_Y}\bar{z}_X(s) + \frac{\lambda_Y}{\lambda_X+\lambda_Y}\bar{z}_Y(s) + 2\frac{\gamma_{XY}(s)}{\lambda_X+\lambda_Y}. \quad (5.40)$$

Proof. Remark that equation (5.38) directly follows from equation (5.37) and $\lambda_U = \pi_1\lambda_X$. Thus, as in Proposition 5.3, we only focus on equation (5.37). By Lemma 5.2 and Proposition 5.3, we have

$$\begin{aligned}
\gamma_{VW}(s) &= \frac{1}{2}\int_{-\infty}^{\infty} k_{VW}e^{-s|t|}dt \\
&= \frac{1}{2}\int_{-\infty}^{\infty}\Big[k_{XY}(t)\left(\pi_1^2 + \pi_0\pi_1 e^{-(\mu_0+\mu_1)|t|}\right) \\
&\qquad\qquad + \pi_0\pi_1 e^{-(\mu_0+\mu_1)|t|}\lambda_X\lambda_Y\Big]e^{-s|t|}dt \\
&= \pi_1^2\gamma_{XY}(s) + \pi_0\pi_1\gamma_{XY}(s+\mu_0+\mu_1) + \frac{\pi_0\pi_1\lambda_X\lambda_Y}{s+\mu_0+\mu_1}.
\end{aligned}$$

□

5.5 Quality of the Peakedness Based Approximation

In this section, we discuss to what extent the characterization of the output flow by its average rate and its peakedness allows us to approximate the PIL for a given routing configuration. As introduced in Section 4.2.2, the idea of the approximation consists in replacing the whole network with a single arc which yields the same average flow rate $\lambda_{out} = \lambda_{X_{out}}$ and peakedness $\bar{z}_{out}(s) = \bar{z}_{X_{out}}(s)$ for a given value of parameter s. From Proposition 5.5, when the network is a single arc with Markov parameters (μ_0, μ_1), the output flow rate and peakedness are given by

$$\lambda_{out} = \pi_1 = \frac{\mu_0}{\mu_0+\mu_1}, \quad (5.41)$$

$$\bar{z}_{out}(s) = \frac{\pi_0}{s+\mu_0+\mu_1} = \frac{\mu_1}{(\mu_0+\mu_1)(s+\mu_0+\mu_1)}, \quad (5.42)$$

since the input flow rate and peakedness are respectively 1 and 0. Hence, for any s, we can write (μ_0, μ_1) as a function of λ_{out} and $\bar{z}_{out}(s)$:

$$\mu_0 = \lambda_{out}\left(\frac{1-\lambda_{out}}{\bar{z}_{out}(s)} - s\right), \tag{5.43}$$

$$\mu_1 = (1-\lambda_{out})\left(\frac{1-\lambda_{out}}{\bar{z}_{out}(s)} - s\right). \tag{5.44}$$

Finally, the PIL is approximated by using the formula for a single arc (see equation (5.12)) with the (μ_0, μ_1) parameters given by equations (5.43-5.44).

Recall that $\frac{1}{s}$ can be interpreted as the average time during which the process is buffered in the peakedness definition. In other words, the parameter s allows us to tune the time scale considered for the peakedness evaluation. Therefore, given FEC parameters (R, K), we naturally choose $s = \frac{1}{T} = \frac{1}{R+K}$, whereas in Section 4.2.2 we obliviously chose $s = 0$. Further, one could argue that, for the purpose of FEC robustness evaluation, the peakedness should be evaluated by buffering the arrived flow according to a time window of exact length T instead of an exponential distribution with mean T, that is one should choose a F-buffer with $F^c(t) = 1$ for $0 \leq t \leq T$ and $F^c(t) = 0$ for $t > T$. However, in that case, the peakedness transformation cannot be described as easily as in Proposition 5.5, which explains why we nevertheless evaluate the peakedness with an exponentially distributed buffering time.

Below we first evaluate the quality of our approximation on instances with two vertices linked by parallel arcs. Then we look at more complicate instances where paths are made of several arcs and are non-disjoint.

5.5.1 Parallel Arcs

We consider here the network model $G = (V, A)$ introduced in Section 5.2.1, where $V = \{v_{in}, v_{out}\}$ and A is a set of arcs going from v_{in} to v_{out}. For these simple instances, we can therefore identify A with the set of paths $\mathcal{P}$. Given splitting ratios $x^{(a)}$ for each $a \in A$, the flow sent on arc a is a constant rate process with arrival rate $x^{(a)}$. For each $a \in A$, the flow received at v_{out} through a is denoted by $X_{out}^{(a)}$ and, from Proposition 5.5, has thus a rate and a peakedness respectively given by :

$$\lambda_{out}^{(a)} = \pi_1^{(a)} x^{(a)},$$

$$\bar{z}_{out}^{(a)}(s) = \frac{\pi_0^{(a)} x^{(a)}}{s + \mu_0^{(a)} + \mu_1^{(a)}}.$$

The output flow X_{out} is given by the superposition of all $X_{out}^{(a)}$, $a \in A$. Hence, by Proposition 5.5 again, X_{out} has a rate and a peakedness respectively given by :

$$\lambda_{out} = \sum_{a\in A} \lambda_{out}^{(a)} = \sum_{a\in A} \pi_1^{(a)} x^{(a)}, \tag{5.45}$$

$$\bar{z}_{out}(s) = = \frac{1}{\lambda_{out}} \sum_{a\in A} \lambda_{out}^{(a)} \bar{z}_{out}^{(a)}(s) \tag{5.46}$$

$$= \frac{1}{\sum_{a\in A} \pi_1^{(a)} x^{(a)}} \sum_{a\in A} \frac{\pi_0^{(a)} \pi_1^{(a)} (x^{(a)})^2}{s + \mu_0^{(a)} + \mu_1^{(a)}}. \tag{5.47}$$

At this point, it is interesting to note that when all arcs have the same stationary probabilities, say $\pi_i^{(a)} = \pi_i$, for all $a \in A$, $i \in \{0, 1\}$, the output arrival rate λ_{out} equals π_1 and is thus independent of the splitting ratios $x^{(a)}$. Therefore, the optimal routing is intuitively obtained by choosing the vector $(x^{(a)})_{a\in A}$ so as to minimize $\bar{z}_{out}(s)$. It is easy to see that the minimum value for $\bar{z}_{out}(s)$ is reached when $(x^{(a)})_{a\in A}$ is a multiple of $(\mu_0^{(a)} + \mu_1^{(a)} + s)_{a\in A}$. We have numerically verified that these ratios are indeed very close to optimal with respect to the exact PIL. Actually this fact can already be observed in the discrete FEC example given in figure 4.8. In this example, the Markov processes have common stationary probabilities $\pi_0 = \frac{1}{101}$ and $\pi_1 = \frac{100}{101}$ and, since each block contains 100 packets, 3 blocks are sent per time unit. Hence $T = 1/3$, $s = \frac{1}{T} = 3$ and $\mu_0 + \mu_1 + s$ on the first and second arcs equal 205 and and 104 respectively. Therefore our method predicts that the flow is optimally split with ratios around $\frac{2}{3} : \frac{1}{3}$, as confirmed in the figure. We have observed the same nice behavior of our approximation method for other instances. However we have not been able to find a theoretical justification.

More generally one can compute the exact probability of irrecoverable loss in the two arc case by a convolution of the distribution for a single arc derived in Section 5.2.3 and, for a large range of parameters, "visualize" the good quality of the peakedness approximation as in figure 4.8. It turns out that, for more than 2 arcs, it is faster to evaluate the PIL by direct Monte Carlo simulation of the Markov processes rather than by a convolution of the loss distribution on each arc. From now on, our approximation is always compared to a probability estimated by simulation. Each simulation has been run sufficiently many times so as to guarantee with probability 0.95 that the estimation is within 10 percent of the exact PIL. These simulations have been run by using basic tools of the Matlab

language.

The 3 arc case is also the last case in which we can easily visualize the probability of irrecoverable loss as a function of the splitting ratio. As an example, let us consider 3 parallel arcs with Markov parameters $(50, 1)$, $(25, 0.5)$ and $(50, 2)$. For FEC parameters $T = 2.5$ and $r = 0.1$ ($R = 0.25$, $K = 2.25$), figures 5.1 and 5.2 respectively show the logarithm (base 10) of the PIL obtained by simulation and the approximated PIL as a function of the splitting ratios on the first and second arc. It immediately appears that our approximation method yields probability values in the same order of magnitude as the "exact" probability obtained from simulations. In particular the optimal point in figure 5.1 is obtained for splitting ratios $0.46 : 0.22 : 0.32$ which yield a minimum PIL of $1.87\ 10^{-5}$. In figure 5.2, the minimum is reached for splitting ratios $0.5 : 0.24 : 0.26$ where the approximated PIL equals $1.05\ 10^{-5}$.

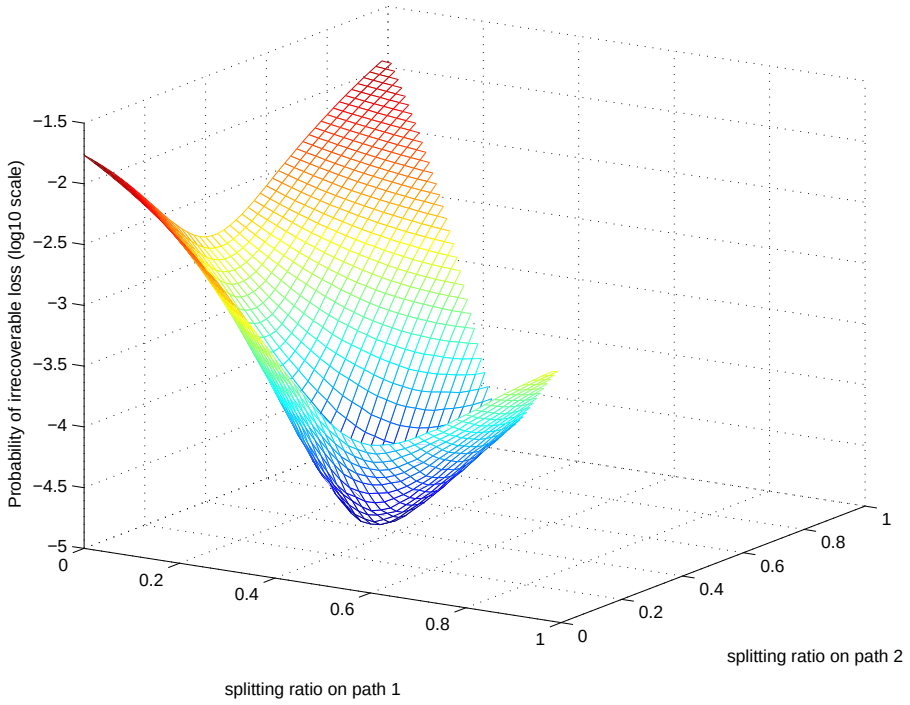

Figure 5.1: *PIL (log10 scale) obtained from simulations for a 3 parallel arcs instance vs splitting ratio on arc 1 and 2*

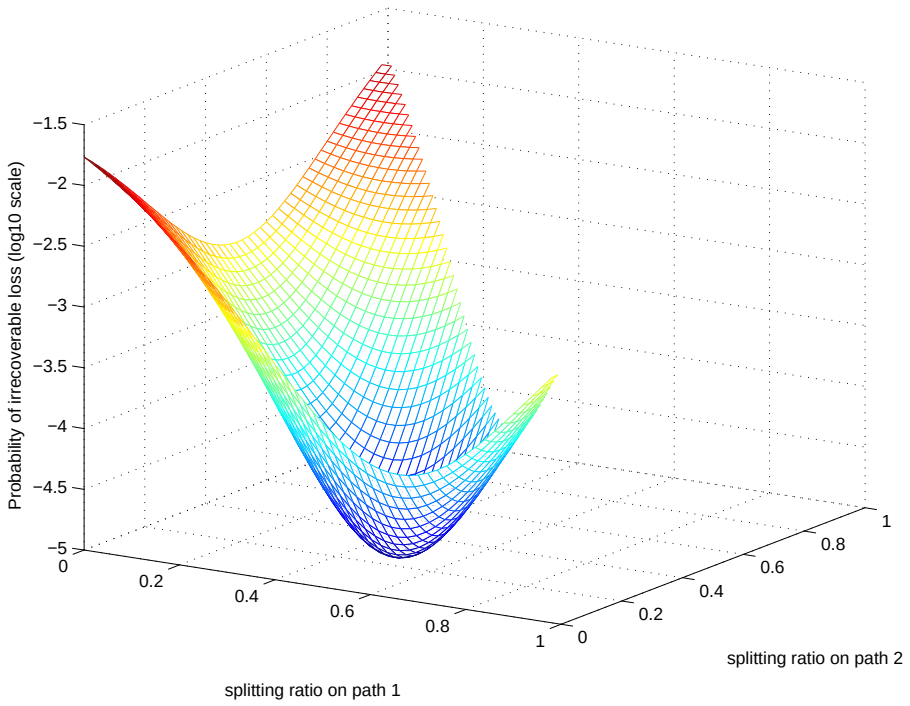

Figure 5.2: *Approximated PIL (log10 scale) for a 3 parallel arcs instance vs splitting ratio on arc 1 and 2*

Even though the relative error between the exact and approximated values of PIL may appear large at first sight, remark that our approximation stays in the right order of magnitude for a large range of probability values. Moreover it seems that this relative error stays stable if we increase the number of parallel arcs.

To justify the previous statement, we have considered network instances with up to 10 parallel arcs. We present here computational results for the 10 arc case. FEC parameters are fixed at $T = 1$ and $r = 0.1$. A set of 100 instances are randomly generated as follows : in each instance, the splitting ratios are randomly chosen in a uniform way, the μ_0 and μ_1 parameters of each arc are uniformly picked in $[25, 50]$ and $[1, 2]$ respectively. These parameter limits are chosen in order to have a reasonable range of probability values: we want the probability values to span several orders of magnitude without reaching extremely small values that would take too much time to estimate accurately. The histogram in figure 5.3 shows how the probability values obtained by simulation are spread on a logarithmic scale.

For each instance, let P_{IL}^S and P_{IL}^A denote the PIL obtained from simulation and from the approximation method respectively. The relative error E is defined by $E = \frac{P_{IL}^S - P_{IL}^A}{P_{IL}^S}$. As announced above, figure 5.4 shows that this error stays within a reasonable range of values. In particular E never exceeds 0.5 for the instances generated.

Finally remark that our approximation always overestimates the PIL. Moreover the quality of the approximation and the order of magnitude of the probability to be estimated seem to be correlated, as depicted in figure 5.5. Therefore a statistical analysis of this dependence could maybe yield a refinement of our method.

5.5.2 More General Topologies

In this section we tackle instances where the paths between v_{in} and v_{out} may have some arcs in common. Hence the flows output on different paths are no longer independent. While it does not essentially change the way we use Monte-Carlo simulations to compute the PIL, it does make our approximation method more complex. Indeed, because of the path dependencies, we must compute the mutual variability between the flows routed on each path in order to compute the peakedness of the final output flow. Moreover, since each path may consist of more than a single arc, the peakedness transformation equations (5.38) require us to compute the

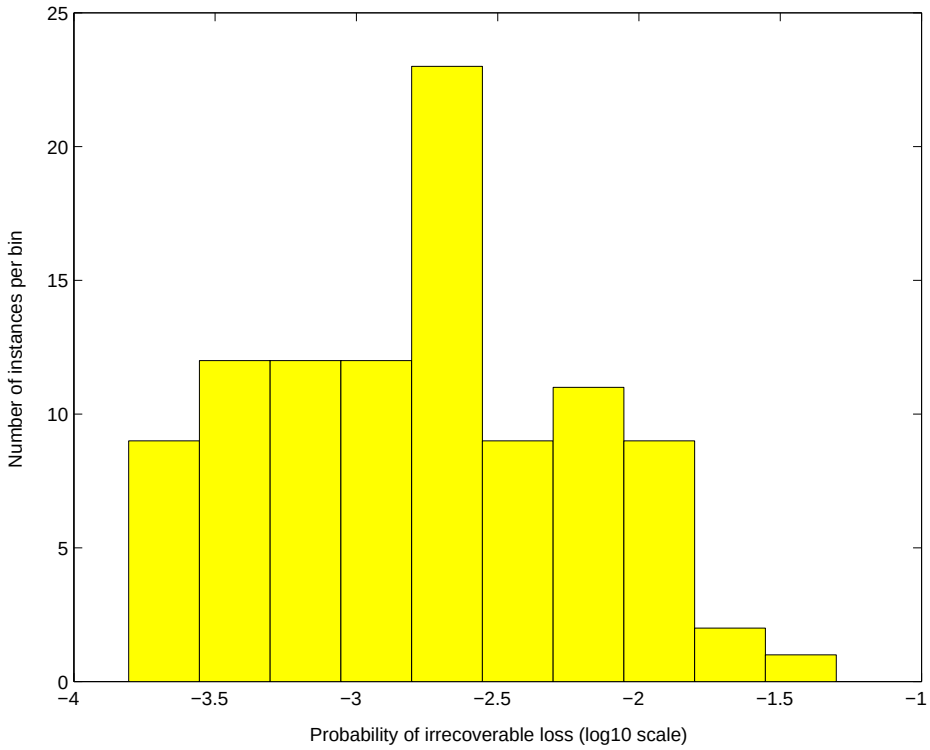

Figure 5.3: *Histogram of the PIL (log10 scale) obtained from simulation for 100 random instances with 10 parallel arcs. The range of PIL values is subdivided into 10 equal bins.*

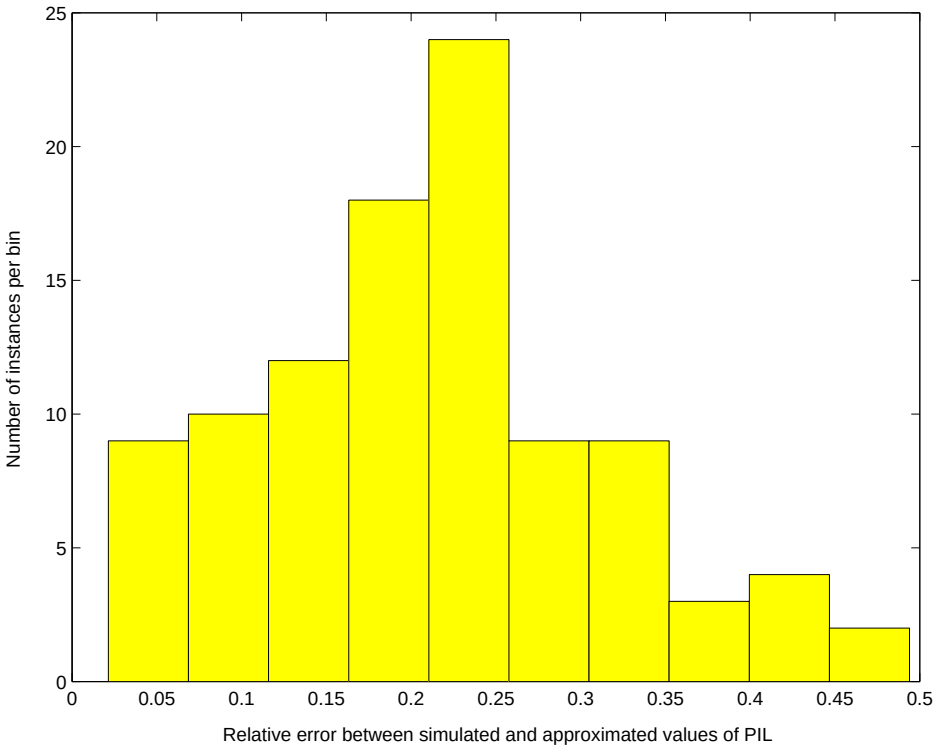

Figure 5.4: *Histogram of the relative error E of the approximated PIL values for 100 random instances with 10 parallel arcs. The range of values for E is subdivided into 10 equal bins.*

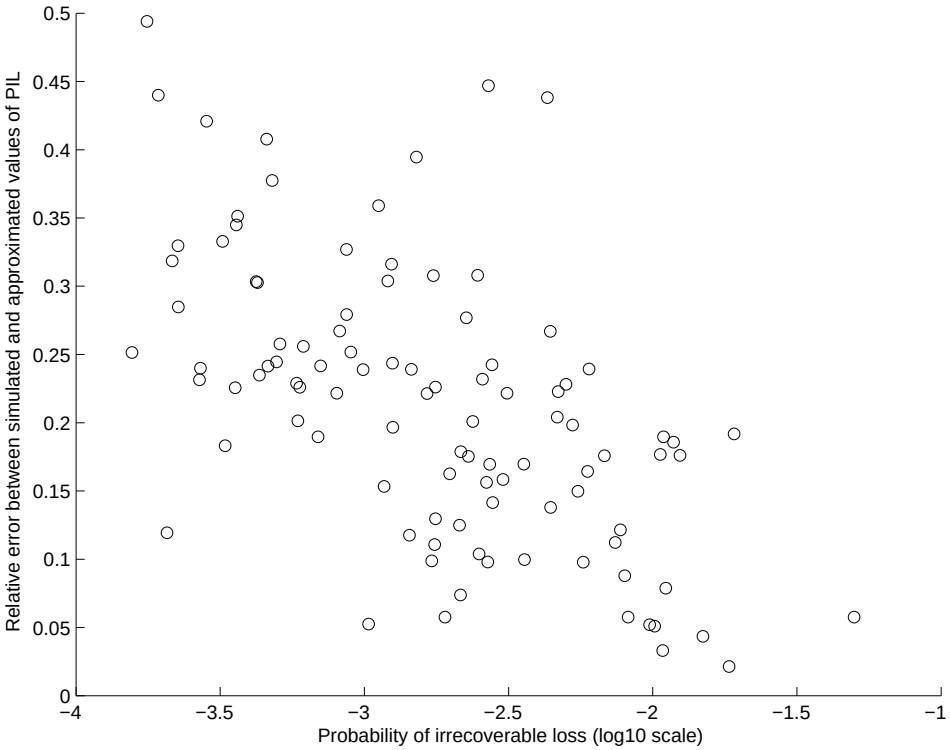

Figure 5.5: *Relative error E vs P_{IL}^{S} (log10 scale) for 100 random instances with 10 parallel arcs.*

peakedness function at different values of the s parameter.

We have tested our approximation method on the network depicted in figure 5.6, which consists of 9 vertices and 12 arcs. One can easily verify that $\mathcal{P}$, the set of $v_{in} - v_{out}$ paths, consists of 6 paths which are clearly not pairwise disjoint.

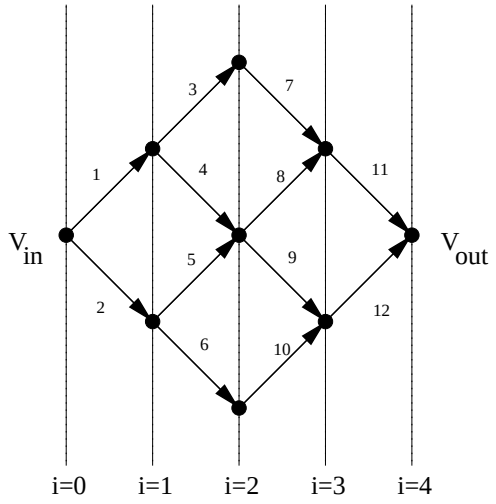

Figure 5.6: *Network instance with 12 non-disjoint paths between v_{in} and v_{out}.*

As in Section 5.5.1, FEC parameters are fixed at $T = 1$ and $r = 0.1$ and a set of 100 instances are generated as follows : in each instance the splitting ratios are randomly chosen in a uniform way, the μ_0 and μ_1 parameters of each arc are uniformly picked in $[50, 100]$ and $[0.25, 1]$ respectively. These values have been chosen so as to obtain the range of probability values shown in figure 5.7. Recall that, for each instance, the value of the PIL is actually obtained by running Monte-Carlo simulations until we obtain an estimation guaranteed to be within 10 percent of the exact value with probability 0.95.

As announced above, in order to compute the final peakedness at v_{out}, we must compute the mutual variabilities between the flows received on each pair of paths in $\mathcal{P}$. For that purpose, we have considered 5 cuts in the networks. These cuts are considered as *stages* labeled from 0 to 4 in figure 5.6. For any $0 \leq i \leq 4$ and $p, q \in \mathcal{P}$, $\lambda_i^{(p)}$ denotes the rate of the flow carried on path p at stage i and $\gamma_i^{(p,q)}(s)$ denotes the mutual variability between the flows carried on path p and q at stage i. Given splitting ratios $x^{(p)}$ for all $p \in \mathcal{P}$, we can sequentially compute the flow rates and variabilities as

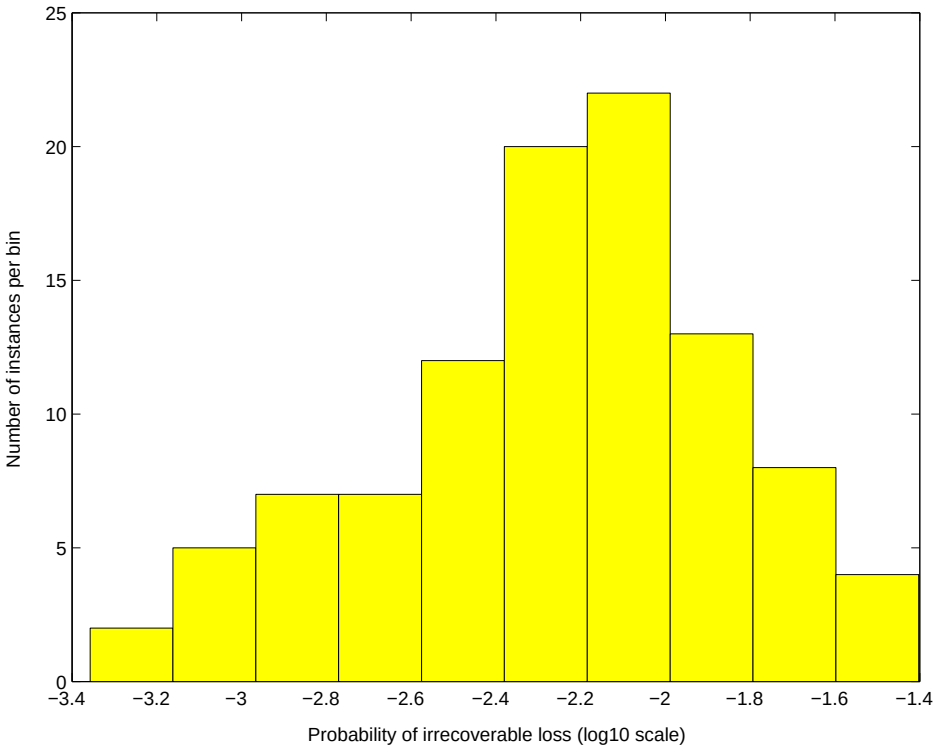

Figure 5.7: *Histogram of the PIL (log10 scale) obtained from simulation for 100 random instances based on the network of figure 5.6. The range of PIL values is subdivided into 10 equal bins.*

follows :

- At stage 0, initialize $\lambda_0^{(p)} := x^{(p)}$ and $\gamma_0^{(p,q)}(s) := 0$
- At stage i, for $1 \leq i \leq 4$, compute $\lambda_i^{(p)}$ and $\gamma_i^{(p,q)}(s)$ from $\lambda_{i-1}^{(p)}$ and $\gamma_{i-1}^{(p,q)}(s)$ by using the transformation equations (5.18) and (5.37) respectively.
- Equations (5.21) and (5.39) allow us to compute the rate λ_{out} and variability $\gamma_{out}(s)$ of the superposition of the flows received in v_{out} respectively as follows :

$$\begin{aligned} \lambda_{out} &= \sum_{p \in \mathcal{P}} \lambda_4^p, \\ \gamma_{out}(s) &= \sum_{p,q \in \mathcal{P}} \gamma_4^{(p,q)}(s). \end{aligned}$$

However, as announced at the beginning of this section, we have to take into account that the variability is actually a function. We can overcome this problem as follows : because of the transformation equations given in Proposition 5.5, one can easily show by induction that the variability between any two flows X and Y in the network instance considered here may be written as

$$\gamma_{XY} = \sum_{j=1}^{J} \frac{\alpha_j}{\beta_j + s},$$

for some positive integer J and positive reals α_j and β_j, for $1 \leq j \leq J$. In our Matlab scripts, we have thus computed the variabilities "symbolically" by actually computing the corresponding sequences $(\alpha_j)_{1 \leq j \leq J}$ and $(\beta_j)_{1 \leq j \leq J}$.

For each instance, we can now compare P_{IL}^S and P_{IL}^A, the values of PIL obtained from simulation and from our approximation method respectively. Figure 5.8 shows that the relative error $E = \frac{P_{IL}^S - P_{IL}^A}{P_{IL}^S}$ has roughly the same range of values than for the parallel arc case. However it seems that P_{IL}^A may underestimate P_{IL}^S. Note that this behavior is maybe explained by the fact that we only compute P_{IL}^S within a 10 percent confidence interval.

Finally we still observe a negative correlation between the value of PIL and the approximation quality, as depicted in figure 5.9. In conclusion,

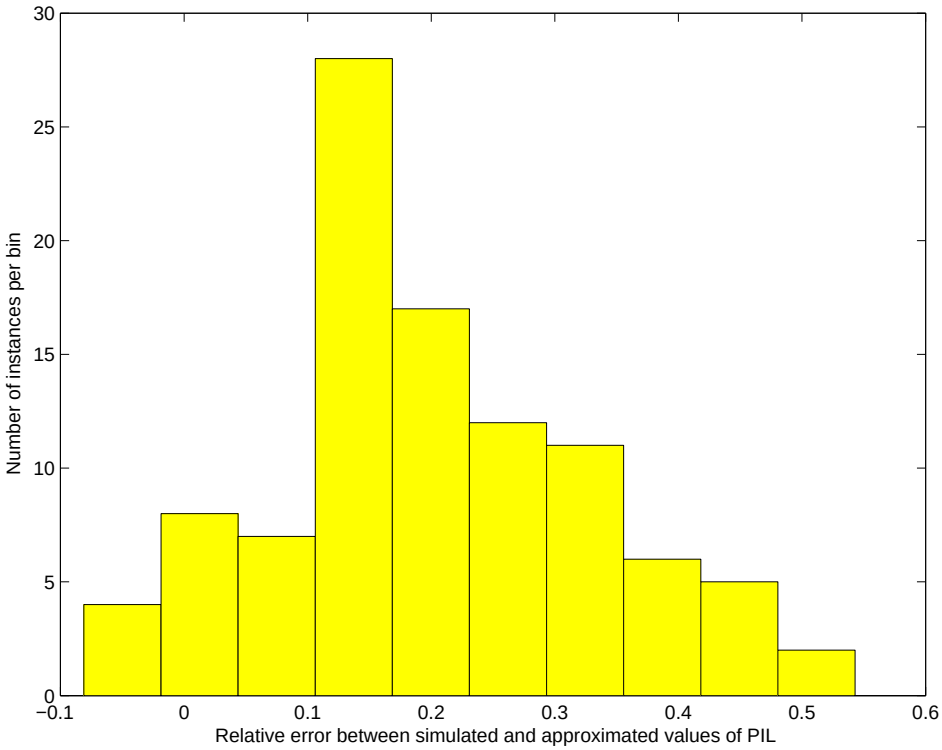

Figure 5.8: *Histogram of the relative error E of the approximated PIL values for 100 random instances based on the network of figure 5.6. The range of values for E is subdivided into 10 equal bins.*

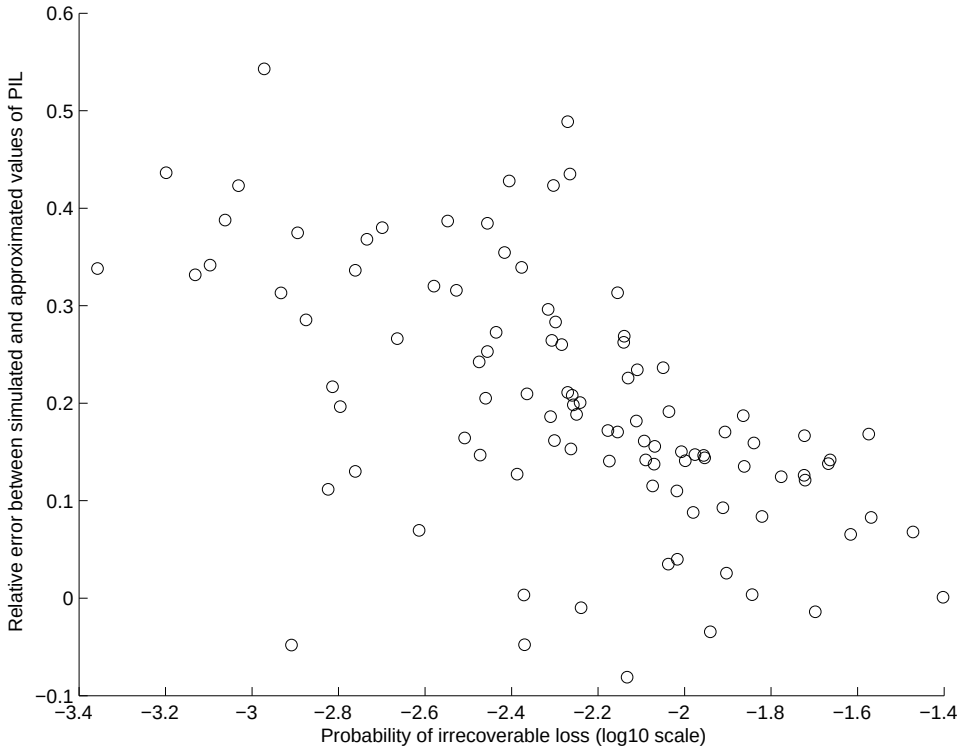

Figure 5.9: *Relative Error E vs P_{IL}^{S} (log10 scale) for 100 random instances based on the network of figure 5.6.*

these computational experiments again show that the peakedness of the output flow allows us to efficiently approximate the PIL of a FEC block for the various instances considered even in the case of non-disjoint paths.

5.6 Conclusion

In this chapter, we have modeled data flows at different points in the network by rate processes. We have presented a network routing model and a fluid interpretation of the FEC protection in this context. Although we have derived an exact formula for the PIL on a single arc, it is numerically challenging to compute the exact PIL on larger network instances. Therefore we have proposed the use of a second-order characterization (its average rate and peakedness) of the flow received at the output node to approximate this probability. For that purpose, we have extended the classical definition of peakedness to arbitrary rate processes and have introduced the notion of mutual variability between flows, which is mandatory to compute the peakedness of the superposition of dependent flows. Finally we have given evidences of the efficiency of our method by comparing the approximated probabilities with values obtained from direct Monte-Carlo simulations. In particular our computational results show that the relative error of our approximation method is reasonably small for a large range of PIL values. Maybe this error could be reduced by using a different buffering distribution to compute the peakedness, as discussed at the beginning of Section 5.5. Moreover, the correlation observed between the value of the PIL and the quality of the approximation raises the hope that statistical techniques can help in improving the efficiency of our method.

Chapter 6

Conclusion and Future Work

In this thesis, we have addressed problems related to the routing of multimedia flows. In the first part, we have considered the problem of finding a core node in a network so as to simultaneously minimize the cost of the unicast and multicast communications between this core node and a set of users. In graph theoretic terminology, this has naturally been formulated as the problem of locating a core vertex in a graph so as to minimize, on the one hand, the sum of the lengths of the shortest paths between the core vertex and and a set of terminal vertices and, on the other hand, the weight of the Steiner tree spanning the core and the terminals. Since this problem is in particular motivated by the minimization of the total bandwidth consumption of applications such as video conferences, we have first considered in Chapter 2 the problem of choosing the core node so as to minimize a weighted sum of both objectives. Computational results show that, in practice, one can quickly find a good core location and that, for many instances, the core is also close to the optimum for each objective regarded independently. These observations were our main motivation to rigorously analyze in Chapter 3 the trade-off between both objectives. In particular, we have derived bounds which guarantee the existence of a core location for which each objective is close to its optimal value (if we were to disregard the other objective).

In the second part of this thesis, we have studied a more complex network model for the purpose of taking into account the interdependency of contiguous data loss events. Such a loss model is mandatory so as to capture the fact that splitting the transmission of a data flow on several paths generally increases the robustness of FEC techniques. In Chapter 4, we have reviewed the modeling approach usually proposed in the litera-

ture and we have discussed some of its limitations. In particular we have proposed to use the peakedness characterization of a flow to approximate the impact of network losses on the FEC robustness. In Chapter 5, a more formal approach has been given by approximating data packet flows by a fluid flow formulation. We have presented a notion of peakedness extended to the fluid context, along with some other tools, and have used them to propose an approximation of the FEC robustness which turns out to be efficient in practice. The main contributions of these two chapters are summarized in the table given at the end of Chapter 4.

Finally let us mention a few directions for future research. In the context of core location problems, we have ignored in this thesis many computational and practical aspects which are essential if one wants to deploy a core selection algorithm on a real data network. A few natural questions are for instance : How to efficiently re-optimize the core location when the users change over time? Can we design a distributed optimization algorithm that could be embedded in a decentralized protocol for the set-up of videoconferences? How do real-time constraints affect the quality of the solution computed? On the theoretical side, it is certainly possible to improve the trade-off results given in Chapter 3. We have actually obtained slightly better results than the one mentioned here with the same type of arguments but at the price of more complicated computations. Therefore we believe that other types of arguments are needed if one wants to eventually close the gap between the trade-off results and the worst known instances.

Concerning the material presented in the second part of the thesis, a very natural question is whether our results helps in finding an optimal set of paths and splitting ratios in order to minimize the probability of irrecoverable loss. Indeed, while we do not see how to tackle this problem by directly considering exact or simulated values of PIL, we believe that using the peakedness and variability transformation equations of Proposition 5.5 may be useful to formulate an approximated but tractable optimization problem. The idea of the formulation is naturally to consider as variables

- the splitting ratios,
- the average rate and variability of the flow on the vertices of each path,
- the average rate λ_{out} and variability $\gamma_{out}(s)$ of the output flow,

and as minimization objective the approximated PIL, which is only a function of λ_{out} and $\gamma_{out}(\frac{1}{T})$ as explained in Section 5.5. These variables are constrained by the equations of Proposition 5.5 which describe how the rate and the variability are transformed while the flow is routed within the network. However there are several issues to address in order to transform this idea into a practical problem formulation that could be solved by generic optimization software. First of all the variability (as the peakedness) is a function and not a a single variable. Therefore it is necessary to define corresponding variables only for a sample of values of the s parameter. A more fundamental issue concerns the convexity of the formulation :

- The objective is in general not a convex function of λ_{out} and $\gamma_{out}(\frac{1}{T})$. Indeed for some instance, we have seen that the the optimal routing is achieved by using a single path even if all paths have the same loss characteristics.
- Equation (5.37), which describes how the variability and the mutual variability are transformed by Markovian losses, does not yield convex constraints. However, for the variability of a single flow (i.e. $X = Y$ and $V = W$ in equation (5.37)), if we replace the equality by an inequality which lower bounds the variability, the obtained constraint is convex. Intuitively this relaxation should not be harmful since, for a given average loss rate, the PIL minimization should seek a routing yielding an output flow with minimum variability. However for the mutual variability of two flows, this trick still yields a non-convex constraint because of the product of flow rates in the last term of equation (5.37).

Since the objective is a function of only two variables, it is likely that its non-convexity can be overcome by explicitly constraining λ_{out} and $\gamma_{out}(\frac{1}{T})$ to lie in an area where the objective is a priori known to be locally convex. Moreover, by focusing on network instances where the flows superposed at the output vertex are independent, we can formulate an approximated optimization problem with convex constraints. Note that, in particular, this is the case for the parallel arc instances considered in Section 5.5.1. For general network topologies, we have to neglect the interdependencies between the flow superposed at the output vertex in order to remove the non-convex constraints on the mutual variabilities. In summary, it seems that some additional assumptions need to be made in order to obtain a

tractable formulation of the problem. Whether these assumptions bring important approximation errors is an interesting question in its own way.

Moreover, many modeling refinements are possible. First of all, a natural extension of the problem consists of introducing arc delays in the model. Then, as already mentioned in Chapter 4, it would be interesting to add capacities in the model and to consider the impact of the network congestion state on its loss characteristics. This would allow us in particular to analyze the global effect of using multipath routing between many different pairs of nodes in the network. Since it uses more network resources than routing on a single shortest path, a potential drawback of multipath routing is to increase the risk of congestion, hence eventually decreasing the quality of the communications. Therefore we find essential to verify whether multipath routing scales well with the total number of transmissions considered. A second type of model improvement consists of replacing the simple Markov loss processes by packet queues of finite size at the entry of each link, in order to model the congestion of router buffers. The challenge is then to extend our approximation results to this more complex network model. A related question concerns the impact of the queuing policies on the peakedness of packet flows. Finally we expect the peakedness characterization to yield efficient and accurate approximations of different types of data flow generated by a large range of applications. We think that many interesting problems could be raised by studying how useful these second-order flow approximations are in order to estimate and optimize the network resources needed to satisfy the quality of service (QoS) required by those applications. In particular, an important question concerns the design and the dimensioning of the network (topology, link capacities, buffer sizes,...) in order to guarantee simultaneously these various QoS with high probability.

Appendix A

List of Publications

A.1 Publications

Journal paper :

- J.-F. Macq and M. X. Goemans, "Trade-offs on the Location of the Core Node in a Network", *Networks*, vol. 44(3), pp. 173-178, October 2004.

Publications in refereed international conferences :

- J.-F. Macq, L.A. Wolsey, and B. Macq, "A Rendez-Vous Point Selection Algorithm for Videoconferencing Applications", in *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME2002)*, vol. 2, pp. 689-692, Lausanne, Switzerland, August 2002.

- J.-F. Macq and M. X. Goemans, "Trade-offs on the Location of the Core Node in a Network", in *Proceedings of the fifteenth annual ACM-SIAM Symposium on Discrete Algorithms (SODA'O4)*, pp. 590-597, New Orleans, Louisiana, USA, January 2004.

- P. Chevalier and J.-F. Macq, "Increasing Forward Error Correction Robustness by Optimizing Multipath Routing", in *Proceedings of the second International Network Optimization Conference (INOC2005)*, vol. 3, pp. 747-752, Lisbon, Portugal, March 2005.

A.2 In Preparation

- P. Chevalier and J.-F. Macq, “Peakedness Based Approximation of the Robustness of Forward Error Correction with Multipath Routing”.

- F.-O. Devaux, B. Macq, and J.-F. Macq, “Scalable Video Multicast Transmissions for Receivers with Different Resolution Requirements”.

Bibliography

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows*, Prentice Hall, Englewood Cliffs, NJ, 1993

[2] K. Andreev, B. M. Maggs, A. Meyerson, and R. K. Sitaraman, "Designing Overlay Multicast Networks For Streaming", in *Proceedings of the fifteenth annual ACM Symposium on Parallel Algorithms and Architectures*, pp. 149 - 158, San Diego, California, USA, 2003.

[3] J. G. Apostopoulos and M. D. Trott, "Path Diversity for Enhanced Media Streaming", *IEEE Communications Magazine*, vol. 42(8), August 2004.

[4] B. Awerbuch, A. Baratz, and D. Peleg, "Cost-Sensitive Analysis of Communication Protocols", in *Proceedings of the 9th ACM Symposium on Principles of Distributed Computing (PODC)*, pp. 177-187, Quebec City, Quebec, Canada, August 1990.

[5] T. Ballardie, P. Francis, and J. Crowcroft, "Core-Based Trees (CBT): An Architecture for Scalable Inter-Domain Multicast Routing", in *Proceedings of ACM SIGCOMM*, pp. 85-95, 1993.

[6] R. E. Bellman, "On a Routing Problem", *Quarterly of Applied Mathematics*, vol. 16, pp. 87-90, 1958.

[7] J. Bolot, S. Fosse-Parisis, and D. Towsley, "Adaptive FEC-Based Error Control for Internet Telephony", in *Proceedings of IEEE INFOCOM 1999*, vol. 3, pp. 1453-1460, 1999.

[8] K. L. Calvert, M. Doar, and E. W. Zegura, "Modeling Internet Topology", *IEEE Communications Magazine*, vol. 35(6), pp. 160-163, June 1997.

[9] K. L. Calvert, E. W. Zegura, and M.J. Donahoo "Core Selection Methods for Multicast Routing", in *Proceedings of the 4th International Conference on Computer Communications and Networks (ICCN'95)*, pp. 638-642, Las Vegas, Nevada, USA, Sept. 1995.

[10] V. Cerf and R. Kahn , "A Protocol for Packet Network Interconnection", *IEEE Transactions on Communications*, vol. 22, pp. 637-648, 1974.

[11] P. Chevalier and N. Tabordon, "Overflow Analysis and Cross-Trained Servers", *International Journal of Production Economics*, vol 85(1), pp. 47-60, Elsevier, July 2003.

[12] Y.-H. Chu, S. G. Rao, and H. Zhang, "A case for End System Multicast", in *Proceedings of ACM SIGMETRICS*, Santa Clara, California, USA, June 2000.

[13] D. D. Clark, "The Design Philosophy of the DARPA Internet Protocols", in *Proceedings of ACM SIGCOMM'88*, pp. 106-114, August 1988.

[14] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*, M.I.T. Press, Cambridge, Massachusetts, U.S.A., 1990.

[15] D. R. Cox, P. A. W. Lewis, *The Statistical Analysis of Series Events*, Methuen & Co, London, 1966.

[16] C. Desset, *Errors and Losses in Image Transmission: Error-Correcting and Recovering Schemes*, Ph.D. Thesis, Université catholique de Louvain, Belgium, 2001.

[17] E. W. Dijkstra, "A Note on Two Problems in Connection with Graphs", *Numerische Mathematik*, vol. 1, pp. 269-271, 1959.

[18] V. A. Ditkin and A. P. Prudnikov, *Operational Calculus in Two Variables and its Applications*, International Series of Monographs on Pure and Applied Mathematics, Pergamon Press, 1962.

[19] M. B. Doar, "A Better Model for Generating Test Networks", in *Proceedings of IEEE Global Internet*, pp. 86-93, IEEE, London, UK, November 1996.

[20] A. E. Eckberg, "A Generalization of Peakedness to Arbitrary Arival Processes and Service Time Distributions", in Bell Laboratories Technical Memorandum, 1976.

[21] A. E. Eckberg, "Generalized Peakedness of Teletraffic Processes", in *Proceedings of the 10-th International Teletraffic Congress*, Montreal, Canada, 1983.

[22] B. M. Edwards, L. A. Giuliano, and B. R. Wright, *Interdomain Multicast Routing: Practical Juniper Networks and Cisco Systems Solutions*, Addison-Wesley, 2002.

[23] E. Fleury, Y. Huang, and P. K. McKinley, "On the Performance and Feasibility of Multicast Core Selection Heuristics, *Networks*, vol. 35(2), pp. 145-156, 2000.

[24] A. Fredericks, "Congestion in Blocking Systems - A Simple Approximation Technique", "The Bell System Technical Journal", vol. 59(6), pp. 805-827, 1980.

[25] V. Frost, "Quantifying the Temporal Characteristics of Network Congestion Events for Multimedia Services", *IEEE Transactions on Multimedia*, vol 5(3), pp. 458-465, IEEE, New-York, Sept. 2003.

[26] R. G. Gallager, *Discrete Stochastic Processes*, Kluwer Academic Publishers, 1996.

[27] R. G. Gallager, P. M. Humblet, and P. M. Spira, "A Distributed Algorithm for Minimum-Weight Spanning Trees", *ACM Transactions on Programming Languages and Systems*, vol. 5(1), pp.66-77, 1983.

[28] I. M. Gel'fand and G. E. Shilov, *Generalized Functions*, vol. 1, Academic Press, New-York, 1964.

[29] I. M. Gel'fand and N. Y. Vilenkin, *Generalized Functions*, vol. 4, Academic Press, New-York, 1964.

[30] M. X. Goemans, Personnal Communication, 1998.

[31] U. Horn, K. Stuhlmuller, M. Link, and B. Girod, "Robust Internet Video Transmission Based on Scalable Coding and Unnequal Error Protection", *Signal Processing: Image Communication*, vol. 15, pp. 77-94, 1999.

[32] F. K. Hwang, D. S. Richards, and P. Winter, *The Steiner Tree Problem*, Elsevier Science Publication B.V., Amsterdam, 1992.

[33] R. M. Karp, "Reducibility among Combinatorial Problems", in *Complexity of Computer Computations*, R. E. Miller and J. W. Thatcher, Eds., pp. 85-103, Plenum Press, New-York, 1972.

[34] S. Keshav, *An Engineering Approach to Computer Networking: ATM Networks, the Internet, and the Telephone Network*, Addison-Wesley, Massachusetts, U.S.A., 1998.

[35] L. Kuang, *Burst Reduction Properties of the Leaky Bucket and the Calculus of Burstiness*, Ph.D. Thesis, University of Maryland, 1992.

[36] D. L. Jagerman, B. Melamed, and W. Willinger, "Stochastic Modeling of Traffic Processes", in J. Dshalalow, editor, *Frontiers in Queueing: Models, Methods and Problems*, pages 271-370, CRC Press, 1997.

[37] M. Kalman, E. G. Steinbach, B. Girod, "Real-Time Voice Communication over the Internet using Packet Path Diversity", in Proceedings ACM Multimedia, pp. 431-440, 2001.

[38] S. O. Krumke, H. Noltemeier, S. S. Ravi and M. V. Marathe, "Bicriteria Compact Location Problems, *Studies in Locational Analysis*, 10, pp. 37-51, Semper Excellens, 1996.

[39] S. Khuller, B. Raghavachari, and N. E. Young, " Balancing Minimum Spanning Trees and Shortest-Path Trees", *Algorithmica*, vol. 14(4), pp. 305-321, 1995.

[40] Y. W. Leung and T. S. Yum, "Connection Optimization for Two Types of Videoconferences, *IEE Proceedings-Communications*, 143, pp. 133-140, 1996.

[41] M. Luby, L. Vicisano, J. Gemmell, L. Rizzo, M. Handley, J. Crowcroft, "The Use of Forward Error Correction (FEC) in Reliable Multicast", RFC 3453, *The Internet Society*, 2002. `http://www.ietf.org/rfc/rfc3453.txt`.

[42] H. Ma and M. Zarki, "Broadcast/Multicast MPEG-2 Video over Wireless Channels using Header Redundancy FEC Strategies", in Proceedings of The International Society for Optical Engineering, pp. 69-90, vol. 3528, 1998.

[43] M. V. Marathe, R. Ravi, R. Sundaram, S. S. Ravi, D. J. Rosenkrantz and H. B. Hunt, "Bicriteria Network Design Problems", *Journal of Algorithms*, vol. 28(1), pp. 142-171, 1998.

[44] B. L. Mark, D. L. Jagerman, and G. Ramamurthy, "Peakedness Measures for Traffic Characterization in High-Speed Networks", in *Proceedings of IEEE INFOCOM 1997*, pp. 427-435, 1997.

[45] K. Melhorn and S. Näher, *LEDA: A Platform for Combinatorial and Geometric Computing*, Cambridge University Press, Cambridge, UK, 2000.

[46] T. Nguyen, "Path Diversity Media Streaming over Best Effort Packet Switched Networks", Ph.D. Thesis, U.C. Berkeley, 2003.

[47] T. Nguyen and A. Zakhor, "Multiple Sender Distributed Video Streaming", *IEEE Transactions on Multimedia*, vol. 6(2), pp. 315-326, 2004.

[48] T. Nguyen and A. Zakhor, "Path Diversity with Forward Error Correction (PDF) System for Packet Switched Networks", in *Proceedings of IEEE INFOCOM 2003*, 2003.

[49] I.S. Reed and G. Solomon, "Polynomial Codes over Certain Finite Fields", *Journal of the Society of Industrial and Applied Mathematics (SIAM)*, vol. 8(2), pp. 300-304, June 1960.

[50] G. Robins and A. Zelikovsky, "Improved Steiner Tree Approximation in Graphs", in *Proceedings of ACM-SIAM Symposium on Discrete Algorithms (SODA2000)*, pp. 770-779, January 2000.

[51] S. Ross, *Stochastic Processes*, J. Wiley & Sons, New-York, 1983

[52] J. Saltzer, D. Reed, and D. Clark, "End-to-end Arguments in System Design", *ACM Transactions on Computer Systems*, vol. 2(4), pp. 195-206, 1984.

[53] W. J. Hopp and M. L. Spearman, *Factory Physics*, Irwin, Chicago, 1996.

[54] C. Stein and J. Wein, "On the Existence of Schedules that are Near-Optimal for Both Makespan and Total Weighted Completion Time", *Operations Research Letters*, 21, pp. 115-122, 1997.

[55] K. W. Stuhlmller, M. Link, B. Girod, and U. Horn, "Scalable Internet Video Streaming With Unequal Error Protection", *Packet Video Workshop*, New York, April 1999.

[56] M. Sudan, "Decoding of Reed Solomon Codes beyond the Error-Correction Bound", *Journal of Complexity*, vol. 13(1), pp. 180-193, 1997.

[57] E. Tardos and K. Wayne, "Simple Generalized Maximum Flow Algorithm", in *Proceedings of the 6th International Integer Programming and Combinatorial Optimization Conference (IPCO'98)*, pp. 310-324, Lecture Notes in Computer Science, vol. 1412, Springer, 1998.

[58] D. Thaler and C. V. Ravishankar, "Distributed Center-Location Algorithms", *IEEE Journal on Selected Areas in Communications*, vol. 15(3), pp. 291-303, 1997.

[59] D. A. van der Laan, *The Structure and Performance of Optimal Routing Sequences*, Ph.D. Thesis, Universiteit Leiden, The Netherlands, 2003.

[60] S. Voss, "Steiner's Problem in Graphs: Heuristics Methods", *Discrete Applied Mathematics*, vol. 40, pp. 45-72, 1992.

[61] R. Wilkinson, "Theories of Toll Traffic Engineering in the U.S.A.", *Bell System Technical Journal*, vol. 35(2), pp. 421-514, 1956.

[62] P. Winter, "Steiner Problem in Networks: a Survey", *Networks*, vol. 17, pp. 129-167, 1987.

[63] R. T. Wong, "A Dual Ascent Approach for Steiner Tree Problems on a Directed Graph", *Mathematical Programming*, vol. 28, pp. 271-287, 1984.

[64] M. Yajnik, S. Moon, J. Kurose, and D. Towsley, "Measurement and Modelling of the Temporal Dependence in Packet Loss", in *Proceedings of IEEE INFOCOM 1999*, vol. 1, pp. 341-352, 1999.