NOTE TO USERS

Université d'Ottawa · University of Ottawa

Université d'Ottawa · University of Ottawa

FACULTÉ DE ÉTUDES SUPÉRIEURES ET POSTDOCTORALES

FACULTY OF GRADUATE AND POSTDOCTORAL STUDIES

Gilles LAMOTHE

AUTEUR DE LA THÈSE - AUTHOR OF THESIS

Ph.D (Mathematics)

GRADE - DEGREE

Department of Mathematics

FACULTÉ, ÉCOLE, DÉPARTEMENT - FACULTY, SCHOOL, DEPARTMENT

TITRE DE LA THÈSE - TITLE OF THE THESIS

Efficient Estimation of the Probability of Extreme Cell Delays

D. McDonald

DIRECTEUR DE LA THÈSE - THESIS SUPERVISOR

CO-DIRECTEUR DE LA THÈSE - THESIS CO-SUPERVISOR

EXAMINATEURS DE LA THÈSE - THESIS EXAMINERS

A. Dabrowski

D. Down

F. Théberge

Y. Zhao

J.-M. De Koninck, Ph.D.

LE DOYEN DE LA FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTDOCTORALES

DEAN OF THE FACULTY OF GRADUATE AND POSTDOCTORAL STUDIES

EFFICIENT ESTIMATION OF THE PROBABILITY OF EXTREME CELL DELAYS

By
Gilles Lamothe, B.Sc., M.Sc.
October 2004

A Thesis
submitted to the Faculty of Graduate and Postdoctoral Studies
in partial fulfillment of the requirements
for the degree of
Doctor of Philosophy in Mathematics[1]

[1]The Ph.D. Program is a joint program with Carleton University, administered by the Ottawa-Carleton Institute of Mathematics and Statistics

Your file Votre référence
ISBN: 0-494-01726-0
Our file Notre référence
ISBN: 0-494-01726-0

Abstract

We wish to estimate the probability of extreme cell delays for a particular virtual connection (VC) or virtual path (VP) in an Asynchronous Transfer Mode (ATM) network. Since extreme cell delays are rare events we cannot efficiently estimate the probability of extreme cell delays using crude Monte-Carlo methods. We increase (exponentially *twist*) the workload of a related Markov additive chain, which we call the encumbrance, and use the A-cycle sampling technique to analyse switches with finite buffers multiplexing cells from Bernoulli interrupted sources with constant bit rate during bursts.

Acknowledgements

I would like to thank my supervisor Prof. David McDonald for all his help, advice and patience. I would like to thank Alcatel, the Ontario Government and the Ottawa-Carleton Institute of Mathematics and Statistics for funding throughout my research.

Dedication

I dedicate this work to my family and friends whom are dear to me.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The technical committee of the ATM Forum (section 5.2 in [ATM96]) specifies that a connection request is established only when sufficient resources are available at each successive network element. The decision to establish the connection through the whole network is based on the service category of the requested connection, the traffic contract, and the quality of service (QoS), and the effect on the agreed QoS of existing connections. The following three QoS parameters are negotiated at the connection request (section 3.1 in [ATM96]), Peak-to-peak Cell Delay Variation (peak-to-peak CDV), Maximum Cell Transfer Delay (maxCTD), and Cell Loss Ratio (CLR).

At the design stage of an ATM switch it is of primordial importance to have estimates of the peak-to-peak CDV, the maxCTD and the CLR for appropriate traffic. Crude Monte-Carlo (CMC) simulation techniques could be utilized. However, with CMC simulations we cannot efficiently estimate these QoS parameters to the elusive and desired order of 10^{-8}. A speed-up (variance reduction) technique is *de rigueur*, e.g. importance sampling, see [Heid95] for a survey, or importance splitting, see [GHSZ96].

In recent years many importance sampling algorithms have been proposed to efficiently estimate the CLR. Chang et al. in [CHJS94] proposed an importance sampling algorithm based on the theory of effective bandwith to estimate cell loss in ATM intree networks. L'Ecuyer and Champoux in [LC96] extended the algorithm from [CHJS94] to include non-intree networks. Falkner et al. in [FDL99] also extended

the algorithm from [CHJS94] using the theory of decoupling bandwidths. Dabrowski et al. in [DLM00] proposed an algorithm based on the h-transform technique (see [McDo99, BDM99]). Bonneau in [Bonn96] used the h-transform technique to efficiently estimate the proportion of non-conforming cells in a leaky bucket controller. Here we concentrate on the CTD.

The end-to-end CTD is the sum of the delays through each of the switching elements. Its upper α quantile is denoted maxCTD, that is

$$\mathrm{P}(\text{end-to-end CTD} > \text{maxCTD}) = \alpha.$$

We say that the area α, for a given upper quantile maxCTD, is the probability of extreme cell delays (PECD). In this paper we propose an importance sampling algorithm based on the h-transform technique to efficiently estimate the PECD for a particular virtual connection (VC) or virtual path (VP). We consider the cells generated by this particular source as tagged cells. We also consider any node through which tagged cells are going to be buffered as a tagged node.

We measure time in time slots of T seconds. The *encumbrance* generated by one cell passing through a tagged node is defined as the reciprocal of the constant link rate at that node. At time slot t, the total encumbrance generated by a cell is defined as the sum of the encumbrance generated at the tagged nodes at which this cell must buffered in subsequent time slots. The total encumbrance in the network at time slot t, ENC_t, is the sum of the encumbrance generated by the cells in the network.

Cells from bursty (Bernoulli interrupted with constant bit rate during bursts – see Section 2.1) sources arrive at the ingress multiplexors and proceed through different sequences of buffered switching points. The cell streams that pass through at least one of the tagged nodes are regarded as cross traffic.

An h-transform or "twist" of the sources causes the total encumbrance, ENC_t, to overload. We simulate the twisted process until the encumbrance reaches a threshold ℓ (usually equal to, or slighty larger than, maxCTD) or the process returns to some "large" subset of states $\mathcal{A}$. We call this a twisted cycle. If and when the total encumbrance reaches ℓ we turn off the twist and count NED, the number of cells which have an end-to-end CTD greater than the maxCTD before the process returns

to $\mathcal{A}$, at which time the cycle ends. We can use NED and an importance sampling weight (likelihood ratio) determined by h (with an additional adjustment for cell loss) to give a consistent estimate of the probability of extreme cell delays. The twist only changes the rates not the routing of cells. Moreover the queueing protocols are not changed. This is desirable in that no new programming is required to simulate the switch.

The $\mathcal{A}$-cycle technique is useful especially when regenerative simulation is not feasible, i.e. returns to a fixed state are prohibitively rare. See [Heid95] for a discussion on the $\mathcal{A}$-cycle technique. This technique assumes that the user can sample from the stationary distribution π on $\mathcal{A}$. Usually this is done approximately using a burn-in technique. In Chapter 7, we show how to sample exactly from a stationary distribution π, when a marginal distribution π_X is known. It is an adaptation of the coupling from the past (CFTP) algorithm from [PW96]. We then give an algorithm to sample exactly from the stationary distribution of a delay process in an ATM-type intree network.

In Chapter 4, we study the large deviations of a two node intree tandem network by applying the theory from [McDo99]. In the Jackson network setting there has been a considerable amount of work done analyzing the large deviations of overflows in two node tandem networks, e.g. [PW89, KN02, GK95]. In Chapter 5, we propose a change of measure for importance sampling (IS) in the more general ATM-type network based on this large deviation result. Such heuristic change of measures for IS-based estimation of probabilities of rare events have been proposed in the past, e.g. [PW89, KN02] in the Jackson network setting or [CHJS94, LC96, FDL99, DLM00] in the ATM-type network setting.

A simulation study for the proposed IS-based estimator is presented in Chapter 6. We estimate the probability of extreme cell delays in a two node tandem network.

We end by giving a giving a "charted map" of the thesis. We try to facilitate the reading of the thesis by providing a general overview of the contents as well as the relationship between the chapters.

A Road Map to the Thesis

1. ATM-Type Networks :

(a) We describe an ATM-type network in Section 2.3.

(b) Our **goal** is to efficiently estimate the probability of extreme cell delays for a tagged virtual connection. We define the "encumbrance" in Section 2.5. The encumbrance is used as a workload process related to the delay process.

(c) In Section 4.2.1, we construct a change of measure using the well-known h-transform technique. Under mild traffic conditions we show the existence of a harmonic function h. To do so we study the spectral properties of the "Feynman-Kac" kernel associated with the Markov additive process with a Markovian part representing the state of the sources and an additive part representing the encumbrance. The spectral results concerning the harmonic function h are found in Lemma 3.4.3 and Lemma 3.6.1.

(d) Using this change of measure, we propose an importance sampling estimator to estimate the probability of extreme cell delays. The description of this estimator is found in Chapter 5. It uses a twin $\mathcal{A}$-cycle technique. Unlike regenerative simulation techniques, there is an initial bias when defining cycles using returns to a set of states $\mathcal{A}$. A burn-in technique is the usual remedy for the so-called initial bias problem. We propose another solution to this problem for a subset of the ATM-type networks, see the discussion below on feedforward intree networks.

2. Feedforward Intree Networks :

(a) Feedforward intree networks are a subset of the ATM-type networks.

(b) In Chapter 7, we adapt the coupling from the past algorithm from [PW96] to sample exactly from the stationary distribution of the delay process. This is a solution to the initial bias problem as discussed above.

3. Two Node Tandem Intree Networks :

(a) Two node tandem intree networks are a subset of the feedforward intree networks.

(b) For this special case, we find sufficient conditions (that are not so mild) to obtain the large deviations of the encumbrance. This is an application of the general theory found in [McDo99]. To do so we had to contruct a Lyapounov function, see Section 4.5.1. Similar to the construction of the harmonic function h, we needed to study the spectral properties of a Feymann-Kac operator. The spectral results concerning the Lyapounov function are found in Lemma 3.5.1 and Lemma 3.6.2.

Chapter 2

General Model

2.1 Sources

It has become common practice to incorporate burstiness when modelling ATM traffic. This can be achieved by using Markov modulated sources. We consider a system with $|A|$ independent sources. Let A be the index set for the sources. Let A_t^a be the number of cells offered by source a to the network during the time slot t. We do not consider batch arrivals, a single source can only offer one (or zero) cell per time slot.

Each source a is modulated by a Markov renewal sequence $(X^a, \hat{M}^a)$, where X^a represents the inter-arrival times and $\hat{M}^a$ is the underlying (irreducible and aperiodic) Markov chain with finite state space, transition kernel $\hat{K}^a$ and stationary distribution $\pi^a_{\hat{M}}$. Define $P^a_{m,m'}$ as the following conditional law

$$P^a_{m,m'}(x) = P(X^a_{k+1} = x | \hat{M}^a_{k+1} = m', \hat{M}^a_k = m).$$

For simplicity, we will assume that the latter distributions are degenerate, i.e.

$$P^a_{m,m'}(c_a(m')) = 1,$$

where $c_a(m)$ will represent the constant inter-arrival times while in state m.

We will allow idle periods, that is periods during which source a does not offer any cells. To model these idle periods, we designate some states as *off*-states, otherwise

say that the state is an *on*-state. If m is an *on*-state, then while the process $\hat{M}^a = m$, then source a will offer a cell every $c_a(m)$ time slots. Denote

$$\iota_a(m) = \begin{cases} 1, & m \text{ is an } on\text{-state}, \\ 0, & m \text{ is an } off\text{-state}. \end{cases}$$

We will assume that the transitions of the underlying Markov chain $\hat{M}^a$ occur at the end of the time slots. For example, assume that the Markov chain, $\hat{M}^a$, jumps into state m at the end of the time slot t. If $\iota_a(m) = 1$, i.e. the source is *on*, then a cell will be offered during the time slot $t + c_a(m)$. The Markov chain then jumps into state m' with probability $\hat{K}^a(m, m')$. This last transition will occur at the end of the time slot $t + c_a(m)$.

Define U^a as the residual number of time slots until the next possible state transition for the underlying Markov chain $\hat{M}^a$. This last process will be useful in order to describe the evolution of the system as a Markov chain. Let M_t^a be the state of source a at the end of time slot t. The process $\{(M_t^a, U_t^a)\}$ is a Markov chain with transition kernel

$$K_A^a(m, u; m', u') = \mathbf{1}_{(u=0)}\, \hat{K}^a(m, m') \mathbf{1}_{(u'=c_a(m')-1)} + \mathbf{1}_{(u>0)}\, \mathbf{1}_{(u'=u-1)}\, \mathbf{1}_{(m=m')}.$$

It follows from Renewal Theory that the stationary distribution for this Markov chain is

$$\pi_a(m, u) = \frac{\pi_{\hat{M}}^a(m)\, \mathbf{1}(u \in \{0, 1, \ldots, c_a(m) - 1\})}{\sum_{m'} \pi_{\hat{M}}^a(m') c_a(m')}.$$

Actually, it is easy to see directly that

$$\sum_{(m,u)} \pi_a(m, u)\, K_A^a(m, u; m', u') = \pi_a(m', u'),$$

and that it is a probability measure, i.e.

$$\begin{aligned} \sum_{(m,u)} \pi_a(m, u) &= \sum_{(m,u)} \frac{\pi_{\hat{M}}^a(m)\, \mathbf{1}(u \in \{0, 1, \ldots, c_a(m) - 1\})}{\sum_{m'} \pi_{\hat{M}}^a(m') c_a(m')} \\ &= \frac{\sum_m \pi_{\hat{M}}^a(m) \sum_u \mathbf{1}(u \in \{0, 1, \ldots, c_a(m) - 1\})}{\sum_{m'} \pi_{\hat{M}}^a(m')\, c_a(m')} \\ &= \frac{\sum_m \pi_{\hat{M}}^a(m)\, c_a(m)}{\sum_{m'} \pi_{\hat{M}}^a(m')\, c_a(m')} = 1. \end{aligned}$$

We define A_t^a as the number of cells offered to the network by source a during the time slot t. The mean number of cells offered per time slot (in steady state) is

$$E_{\pi_a}[A_t^a] = \sum_m \pi_a(m,0)\,\iota_a(m),$$

or

$$E_{\pi_a}[A_t^a] = \frac{\sum_m \pi^a_{\hat{M}}(m)\,\iota_a(m)}{\sum_m \pi^a_{\hat{M}}(m)\,c_a(m)}. \tag{2.1.1}$$

We will partition the index set A into the sets T_{vc}, C_{vc} and R_{vc} : the tagged cell streams, the cross traffic, and the remainder of the traffic streams, respectively. Explanations are given in Section 2.5. We end the section with two examples.

Example 2.1.1 (CBR Sources) *Source a is said to be constant bit rate(CBR), if it offers a cell periodically, let us say every κ_a time slots. In the above framework, the distributions of the inter-arrival times and the transitions are degenerate. We can view $\hat{M}^a$ as a Markov chain with the degenerate state space $\{1\}$ and the support of the distribution of the inter-arrival times is $\{\kappa_a\}$, i.e. $c_a(1) = \kappa_a$. Its stationary distribution is*

$$\pi_a(1,u) = 1/\kappa_a \quad \text{for } u \in \{0,1,\ldots,\kappa_a - 1\},$$

and the mean number of cells offered per time slot (in steady state) is

$$E_{\pi_a}[A_t^a] = \pi_a(1,0) = \frac{1}{\kappa_a}.$$

Example 2.1.2 (VBR Sources) *It is common in the literature, see [LC96, FDL99, DLM00], to model a variable bit rate (VBR) source as an on-off process with geometric sojourn times. Such a source has an embedded alternating renewal process, where a cycle consists of an idle and a burst period.*

Assume source a to be an on-off source. The sojourn times of the idle periods are geometric with parameter p^a_{01}. During a burst period, source a will offer a cell to the network every κ_a time slots. This allows us to model sources with different peak cell rates. The size of a burst is assumed to be geometric with parameter p^a_{10}.

In the above framework, taking on $\equiv 1$ and off $\equiv 0$, we obtain the following:

$$c_a(1) = \kappa_a, \quad \text{and} \quad c_a(0) = 1.$$

The transition kernel for the underlying Markov chain is

$$\hat{K}^a = \begin{pmatrix} p^a_{00} & p^a_{01} \\ p^a_{10} & p^a_{11} \end{pmatrix}.$$

From renewal theory, the stationary distribution for (M^a, U^a) *can be easily shown to be given by*

$$\pi_a(1,u) = \frac{1/p^a_{10}}{\kappa_a/p^a_{10} + 1/p^a_{01}} \mathbf{1}(u \in \{0,1,\ldots,d^a_A - 1\})$$

and

$$\pi_a(0,0) = \frac{1/p^a_{01}}{\kappa_a/p^a_{10} + 1/p^a_{01}}.$$

The mean number of cells offered per time slot (in steady state) is

$$E_{\pi_a}[A^a_t] = \pi_a(1,0) = \frac{1/p^a_{10}}{\kappa_a/p^a_{10} + 1/p^a_{01}}.$$

2.2 Services

Take B as the index set for the nodes in the network. We assume that the services are deterministic and that every node is work conserving. Suppose that the time slot t is a possible service epoch. If, at the end of the time slot $t-1$, there was at least one cell buffered at the node b, then a cell is served. The buffer at node b will have a finite capacity L^b.

If Q^b represents the number of cells buffered at the node b, then it satisfies the following recurrence equation:

$$Q^b_t = \max\{A_t(b) + Q^b_{t-1}, L^b\} - \mathbf{1}\{Q^b_{t-1} > 0, V^b_{t-1} = 0\},$$

where V^b is the residual number of time slots until the next service slot and $A_t(b)$ is the number of cells offered to the node b during the tth time slot. We assume that there is a service slot at every d_b time slots, i.e.

$$V^b_t = \begin{cases} d_b - 1, & \text{if } V^b_{t-1} = 0 \\ V^b_{t-1} - 1, & \text{otherwise.} \end{cases}$$

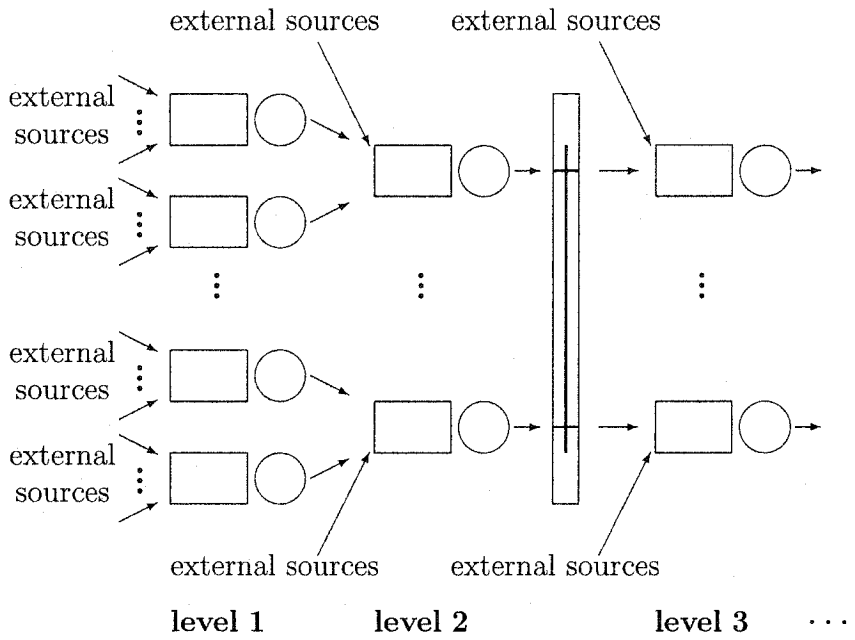

Figure 1: ATM-Type Network

2.3 ATM-Type Queueing Network and Associated Queuing Processes

We consider an ATM-type queueing network as in [FDL99, LC96]. It is a feedforward network of *work conserving* nodes with finite buffers. The nodes are partitioned into levels, see Figure 1. We assume a fixed number of levels, say r. The first two levels are assumed to be *intree*. The departing stream of cells from a second (or higher) level node can split to allow the cells from different *virtual connections* (sources) to be routed appropriately in order to reach their respective destinations. We will assume fixed routing. All cells offered by a particular source are going to follow the same path through the network.

We will allow a source to offer its cells to the network at any fixed node in the network. This means, there can be external sources at all levels.

We need to consider a large enough state space in order to describe the routing of the cells through this network. This can be done using a multiclass framework similar

to [RS92, Dai95]. All cells from the same class departing a fixed node are going to be routed to the same node in the next level. Once a cell is routed it changes class. Since there is a finite number of nodes and thus paths though the network, there is also a finite number of classes. Let $\mathfrak{C}$ be the index set for the classes. Denote $\mathfrak{c}_t^b(i) \in \mathfrak{C}$ as the class of the ith cell buffered at the node b at the end of the time slot t. The class 0 will be used for an empty slot, that is $\mathfrak{c}_t^b(i) = 0$, if $Q_t^b < i$. Denote $\mathfrak{c}_t^b = (\mathfrak{c}_t^b(1), \ldots, \mathfrak{c}_t^b(L_b))$ and let $\{f_b\}_{b\in B}$ be a collection of $\mathbb{Z}_+$-valued functions on $\mathfrak{C}$. In our case, the quantity $f_b(\mathfrak{c}_t^b(i))$ will represent the encumbrance remaining in the system at the end of the time slot t due to the ith cell buffered at node b. We introduce the encumbrance in Section 2.5.

We are interested in the large deviations of queuing processes associated to this network, e.g. the workload of a hot spot buffer (see [DLM00]), or the encumbrance defined in Section 2.5. These large deviations results will help us to construct an efficient importance sampling estimator for the probability of certain rare events. The queueing processes of interest are going to be of the form

$$Z_t = \sum_{b\in B} Z_t^b, \quad \text{where} \quad Z_t^b = \sum_{i=1}^{L_b} f_b\left(\mathfrak{c}_t^b(i)\right). \tag{2.3.1}$$

Example 2.3.1 (Workload) *In [DLM00], the workload of the hot spot buffer is defined as the number of cells in the network which are currently (or are going to be during some future time slot) buffered in a particular hot spot buffer. This process is clearly of the form (2.3.1).*

2.4 Delay Process

The state space of the Markov chain $\{Z_t, (\mathfrak{c}_t^b, V_t^b)_{b\in B}, (M_t^a, U_t^a)_{a\in A}\}$ is large enough to describe the evolution of the queueing process Z_t. But, it is not large enough to describe the evolution of the delay process. Since we are interested in the end-to-end CTD for tagged cells, we must consider an even larger state space.

Denote $D_t^b(i) \in S_D \subset \mathbb{Z}_+$ as the age of the ith cell buffered at the node b at the end of the time slot t, where S_D is the set of possible delays. Note, S_D is a finite set since we are considering buffers of finite capacity.

Let $D_t^b = (D_t^b(1), \ldots, D_t^b(L_b))$. The process $\{Z_t, (D_t^b, \mathfrak{c}_t^b, V_t^b)_{b\in B}, (M_t^a, U_t^a)_{a\in A}\}$ is a Markov chain. We assume the existence of a stationary distribution $\breve{\pi}$ for this Markov chain. In Chapter 5, we present an importance sampling algorithm to efficiently estimate the probability of extreme cell delays.

2.5 The Encumbrance

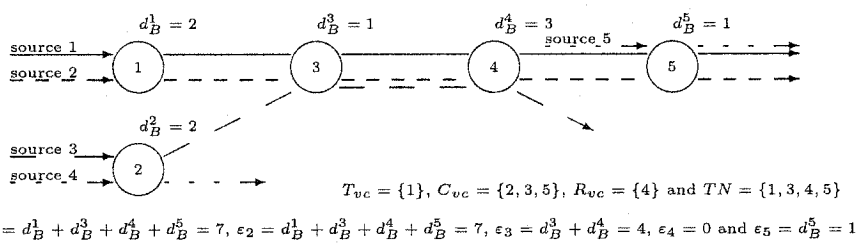

Figure 2: An ATM-Type Network

We want to estimate the upper-tail probability associated with the upper-quantile maxCTD for the tagged cells. All nodes in the path of the tagged cells are called tagged nodes. At the end of time slot t, a distinct non-negative integer c is attached to each cell. Define $TN(c) \subset TN$ as the set of tagged nodes that still need to be traversed by the cell c in the network. We call the inter-service time d_b, the ***encumbrance*** of a cell at node b. The ***total encumbrance*** at the end of time slot t is defined as follows:

$$\mathrm{ENC}_t^{\breve{}} := \sum_c \sum_{b\in TN(c)} d_b. \tag{2.5.1}$$

Note, when a cell is served at a tagged node b, the total encumbrance is reduced by d_b.

We will partition the index set A of the sources into the sets T_{vc}, C_{vc} and R_{vc} : the tagged cell streams, the cross traffic, and the remainder of the traffic streams, respectively. For an example, see Figure 2. The cross traffic are the non-tagged cells with at least one tagged node in their path. The remainder of the traffic represents

the cells that do not pass through a tagged node. It is only the tagged cell streams and the cross traffic that contribute to the total encumbrance.

Our goal is to exponentially twist the total encumbrance such that "on average" the total encumbrance will tend to grow. As the total encumbrance grows then the queues also grow which should cause longer delays. This twist is then used to construct an importance sampling estimator for the upper-tail probability associated with the upper-quantile maxCTD for the tagged cells.

To construct the change of measure, we ignore the reflections due to finite buffers, i.e. we will assume that all buffers have an infinite capacity. To denote that a process is associated to the finite buffers regime, we will utilise the superscript $\breve{}$. For example, ENC represents the total encumbrance in the infinite buffers regime, while $\breve{\text{ENC}}$ the total encumbrance in the finite buffer regime.

When considering infinite buffers, the number of cells buffered at node b, Q^b will satisfy the following recurrence

$$Q^b{}_t = A(b)_t + Q^b{}_{t-1} - \mathbf{1}\{Q^b{}_{t-1} > 0, V^b{}_{t-1} = 0\}.$$

The process $\{ENC_t, (\mathfrak{c}^b{}_t, V^b{}_t)_{b\in B}, (M^a{}_t, U^a{}_t)_{a\in A}\}$ is a Markov chain. We denote its transition kernel as K. We assume the existence of a stationary distribution π for this Markov chain. We denote its state space as S. A general vector from the state space S is denoted $w = (e, \vec{\mathfrak{c}}, \vec{v}, \vec{m}, \vec{u})$.

In Section 4.3, we define the set of states $\Delta \subset S$ such that it includes the states for which at least one of the tagged nodes is idle, i.e.

$$\{w \in S : \text{ there exists a } b \in TN, \mathfrak{c}^b = \mathbf{0}\} \subseteq \Delta,$$

where $\mathfrak{c}^b = \mathbf{0}$ means that there are no cells buffered at the node b. Away from Δ, the increment for the total encumbrance is

$$\Delta\, ENC_t = \sum_{a\in T_{vc}\cup C_{vc}} \varepsilon_a\, \mathbf{1}\{\text{cell offered by source } a\} - \sum_{b\in TN} d_b\, \mathbf{1}\{V^b_{t-1} = 0\}, \quad (2.5.2)$$

where ε_a is the encumbrance due to a cell offered by source a as it enters the system and Δ is the difference operator defined as $\Delta\, X_t = X_t - X_{t-1}$. Recall, the encumbrance

due to a tagged cell (when it enters the system) is equal to the sum of the inter-service times d_b over all tagged nodes b along its path.

We will also need an associated free process to construct the change of measure. A detailed explanation of the general method to obtain the change of measure is given in Chapter 4. We construct the free process ENC^∞ by ignoring the reflections at Δ. The superscript ∞ will indicate this regime. The increment generating the total encumbrance under this regime is defined as

$$\Delta ENC_t^\infty = \sum_{a \in T_{vc} \cup C_{vc}} \varepsilon_a \, \mathbf{1}\{\text{cell offered by source } a\} - \sum_{b \in TN} d_b \, \mathbf{1}\{Y^b{}_{t-1} = 0\}. \quad (2.5.3)$$

We simply took the increment for ENC away from Δ, see (2.5.2), as the definition for ΔENC_t^∞. It should be noted that under this regime the total encumbrance ENC^∞ can take on negative values. Hence, we must extend the state space S to allow these negative values. The extended state space will be denoted S^∞.

We call the process $\{ENC_t^\infty, (\mathfrak{c}^b{}_t, V^b{}_t)_{b \in B}, (M^a{}_t, U^a{}_t)_{a \in A}\}$ the *free* process. It is a Markov chain evolving on the state space S^∞. We denote its transition kernel as K^∞. The state space S^∞ is divided into three regions, the edge $\Delta \subset S$, the interior $S^\circ := S \setminus \Delta$ and the boundary $\blacktriangle := (S^\infty \setminus S) \cup \Delta$. The free process has been constructed in such a way that the kernels K and K^∞ agree in the interior S°, that is

$$K(w; w') = K^\infty(w; w') \quad \text{for all } w, w' \in S^\circ.$$

We introduce notation and terminology that will be used in Chapter 4. Let $W^\infty = (\tilde{W}^\infty, \hat{W}^\infty)$ be given by

$$\tilde{W}^\infty[t] = \left(ENC_t^\infty, (\mathfrak{c}^b{}_t)_{b \in B}\right) \quad \text{and} \quad \hat{W}^\infty[t] = \left((V^b{}_t)_{b \in B}, (M^a{}_t, U^a{}_t)_{a \in A}\right).$$

The process $\hat{W}^\infty$ is called the *Markovian part* and the first component of $\tilde{W}^\infty$, that is $\tilde{W}_1^\infty = ENC^\infty$ is called the *additive part*. This terminology stems from the fact that the marginal process $(\tilde{W}_1^\infty, \hat{W}^\infty)$ is a Markov additive process. This means that $(\tilde{W}_1^\infty, \hat{W}^\infty)$ is a Markov chain whose transition kernel, denoted $\mathcal{J}$, has the form

$$\mathcal{J}(\tilde{w}_1, \hat{w}; \tilde{w}_1', \hat{w}') = \mathsf{P}\left[\Delta \tilde{W}_1^\infty[1] = \tilde{w}' - \tilde{w}, \hat{W}^\infty[1] = \hat{w}' | \hat{W}^\infty[0] = \hat{w}\right].$$

Actually in our case, conditioned on $\tilde{W}_1^\infty[t] = ENC_t^\infty = e$, and $\hat{W}[t] = \hat{w}$ the random variable $W_1^\infty[t] = ENC_t^\infty$ is almost surely constant. This follows from our construction. We denote this constant $\varsigma(e, \hat{w})$. Furthermore, conditioned on $\hat{W}[t] = \hat{w}$, the increment of the total encumbrance is almost surely a constant. In other words, for fixed $\hat{w}$, the difference $\varsigma(e, \hat{w}) - e$, is a constant for all e. We conclude that the kernel $\mathcal{J}$ has the following form,

$$\mathcal{J}(e, \hat{w}; e', \hat{w}') = \mathbf{1}(e' = \varsigma(e, \hat{w}))\, \mathsf{P}\left[\hat{W}^\infty[1] = \hat{w}' | \hat{W}^\infty[0] = \hat{w}\right]. \tag{2.5.4}$$

We can consider $\tilde{W}_1^\infty = ENC^\infty$ as a random walk on the additive group $\mathbb{Z}$ with increments that are modulated by the Markovian part $\hat{W}^\infty$ (sometimes called the environment). It should be natural to suspect the existence of an exponential harmonic function h for this random walk.

In Section 4.2, we will construct a harmonic function h for the transition kernel K^∞ of the form $h(\tilde{w}_1, \hat{w}) = \exp(\theta \tilde{w}_1)\, \hat{a}(\hat{w})$, where θ is a real positive constant and $\hat{a}(\cdot)$ is a real positive function. We say that h is harmonic for K^∞ if it satisfies the averaging property

$$\sum_{w' \in S^\infty} K^\infty(w; w')\, h(w') = h(w), \qquad \text{for all } w \in S^\infty. \tag{2.5.5}$$

The change of measure or *twist* is constructed using the harmonic function h. The h-transform for K^∞ is

$$\mathcal{K}^\infty(w; w') := K^\infty(w; w')\, \frac{h(w')}{h(w)}.$$

It is a direct consequence of the averaging property (2.5.5), that $\mathcal{K}^\infty$ is a genuine probability transition kernel, i.e. $\sum_{w'} \mathcal{K}^\infty(w, w') = 1$, for all $w \in S^\infty$. The process $\mathcal{W}^\infty$ generated by the transition kernel $\mathcal{K}^\infty$ is called the twisted free process.

In Chapter 5, we will use this h-transform to construct an importance sampling estimator for the probability of extreme cell delays.

Chapter 3

Spectral Theory

Consider an ATM-type source a as defined in Section 2.1. In this chapter, we investigate the spectral theory of the following operator

$$\widehat{K}_M^a(m,m'|\alpha_a,\gamma) := e^{[\alpha_a\, c_a(m')+\varepsilon_a\,\gamma\,\iota_a(m)]}\,\hat{K}^a(m,m'), \tag{3.0.6}$$

where ε_a is the encumbrance due to a cell offered by source a as it enters the system, $c_a(m')$ is the inter-arrival rate in state m' and $\iota_a(m)$ is one or zero if m is an *on* or an *off* state, respectively.

The results found here are similar to those found in [BDM99]. Although, since we assumed that the Markov chain $\hat{M}^a$ has a finite state space then we will not require the general spectral theory from [NN87]. The operator $\widehat{K}_M^a$ is a non-negative square matrix, thus we can make use of the well-known Perron-Frobenius Theorem.

We end the chapter, by showing similar results for the nodes.

3.1 Degenerate Matrix - CBR Sources

Consider the CBR sources of Example 2.1.1. For each CBR source a, the operator $\widehat{K}_M^a$ is a degenerate one-dimensional matrix, i.e.

$$\widehat{K}_M^a(1,1|\alpha_a,\gamma) = \exp\left[\alpha_a\,\kappa_a + \varepsilon_a\,\gamma\right],$$

where κ_a is the constant inter-arrival time. It has the degenerate eigenvalue

$$\rho_a(\alpha_a,\gamma) = \exp\left[\alpha_a\,\kappa_a + \varepsilon_a\,\gamma\right],$$

with associated eigenmeasure $l_a(1) = 1$ and eigenfunction $r_a(1) = 1$.

For $\gamma \in \mathbb{R}$, we will define $\alpha_a(\gamma)$ as the unique α_a that satisfies $\rho_a(\alpha(\gamma), \gamma) = 1$. Hence,

$$\alpha_a(\gamma) = -\frac{\varepsilon_a \gamma}{\kappa_a}.$$

It follows directly that $-\alpha_a(\cdot)$ is a differentiable, convex function, $\alpha_a(0) = 0$, and

$$\frac{d}{d\gamma}\{-\alpha_a(\gamma)\}\Big|_{\gamma=\gamma} = \frac{\varepsilon_a}{\kappa_a} = \varepsilon_a E_{\pi_a}[A_t], \quad \text{for all } \gamma \in \mathbb{R},$$

where $E_{\pi_a}[A_t]$ is the mean number of cells offered per time slot by source a.

3.2 Non-Degenerate Matrix - Bursty Sources

Consider a non-degenerate ATM-type source a of Section 2.1. The underlying Markov chain $\hat{M}^a$ is assumed to be irreducible and aperiodic. Thus, the matrix $\widehat{K}^a_M$ as defined by (3.0.6) is an irreducible and aperiodic matrix. Thus, its Perron-Frobenius eigenvalue $\rho_a(\alpha_a, \gamma)$, i.e. maximal eigenvalue, is a z-root of order 1 of the polynomial equation

$$\det(\widehat{K}^a_M - z\, I) = 0.$$

Therefore, by the Implicit Function Theorem for several complex variables, $\rho(\alpha_a, \gamma)$ is a holomorphic function, see [GR65].

The Perron-Frobenius Theorem also tells us that the matrix $\widehat{K}^a_M$, has a unique positive eigenmeasure l_a and positive eigenfunction r_a associated to the Perron root ρ_a such that

$$\sum_m l_a(m)\, r_a(m) = 1. \tag{3.2.1}$$

Note that $\rho_a = 1$, $r_a \equiv 1$ and $l_a = \pi^a_{\hat{M}}$, when $(\alpha_a, \gamma) = (0, 0)$, since $\widehat{K}^a_M = \hat{K}^a$, which is a stochastic matrix.

Since $c_a(m') \geq 1$ for all m', that is the inter-arrival times must be at least one time slot, then it should be evident that for fixed γ, $\rho(\alpha_a, \gamma)$ is a strictly increasing function in α_a and that

$$\lim_{\alpha_a \to +\infty} \rho(\alpha_a, \gamma) = +\infty \quad \text{and} \quad \lim_{\alpha_a \to -\infty} \rho(\alpha_a, \gamma) = 0.$$

Thus, there exist a unique real number α_a such that $\rho_a(\alpha_a, \gamma) = 1$. We can conclude that the function $\alpha_a(\gamma)$ defined implicity by

$$\rho_a(\alpha_a(\gamma), \gamma) = 1, \tag{3.2.2}$$

is well-defined and exists for all $\gamma \in \mathbb{R}$. This function is differentiable by the Implicit Function Theorem.

For the remainder, we will assume that α_a satisfies (3.2.2), which will imply that $\rho_a = 1$. We obtain

$$\sum_{m'} \widehat{K}_M^a(m, m')\, r_a(m') = r_a(m),$$

and

$$\sum_{m} l_a(m)\, \widehat{K}_M^a(m, m') = l_a(m').$$

From these two equations and (3.2.1), we also obtain

$$\sum_{m} \sum_{m'} l_a(m)\, \widehat{K}_M^a(m, m')\, r_a(m') = 1.$$

We will differentiate both sides of this equation to obtain the $(\alpha_a)'$, i.e. the derivative of α_a with respect to γ. The derivative of the right-hand-side is zero. Note, we denote the derivatives of l_a and r_a as l_a' and r_a', respectively. The derivative of the left-hand-side is the sum of the following three terms,

$$\begin{aligned} &\sum_{m} \sum_{m'} l_a'(m)\, \widehat{K}_M^a(m, m')\, r_a(m') + \sum_{m} \sum_{m'} l_a(m)\, \widehat{K}_M^a(m, m')\, r_a'(m') \\ + &\sum_{m} \sum_{m'} l_a'(m)\, \frac{d}{d\gamma} \widehat{K}_M^a(m, m')\, r_a(m'). \end{aligned}$$

Sum over m' in the first term, sum over m in the second term and differentiate in the third term, to obtain

$$\begin{aligned} &\sum_{m} l_a'(m)\; r_a(m) + \sum_{m'} l_a(m')\, r_a'(m') \\ + &\sum_{m} \sum_{m'} l_a(m)\; \left[(\alpha_a)'\, c_a(m') + \varepsilon_a\, \iota_a(m)\right]\; \widehat{K}_M^a(m, m')\, r_a(m'), \end{aligned}$$

since

$$\begin{aligned}\frac{d}{d\gamma}\widehat{K}^a_M(m,m') &= \frac{d}{d\gamma}\left\{e^{[\alpha_a\, c_a(m')+\varepsilon_a\,\gamma\,\iota_a(m)]}\,\hat{K}^a(m,m')\right\}\\ &= [(\alpha_a)'\,c_a(m')+\varepsilon_a\,\iota_a(m)]\;e^{[\alpha_a\, c_a(m')+\varepsilon_a\,\gamma\,\iota_a(m)]}\,\hat{K}^a(m,m')\\ &= [(\alpha_a)'\,c_a(m')+\varepsilon_a\,\iota_a(m)]\;\widehat{K}^a_M(m,m').\end{aligned}$$

Note that $\sum l'_a\, r_a + \sum l_a\, r'_a = 0$, since the left-hand-side is the derivative of $\sum l_a\, r_a$ which is equal to one. This implies that the remaining term must also be equal to zero, that is

$$\sum_m \sum_{m'} l_a(m)\;[(\alpha_a)'\,c_a(m')+\varepsilon_a\,\iota_a(m)]\;\widehat{K}^a_M(m,m')\,r_a(m') = 0.$$

Isolating for $-(\alpha_a)'$, we obtain

$$-(\alpha_a)' = \frac{\sum_m \sum_{m'} l_a(m)\,\varepsilon_a\,\iota_a(m)\,\widehat{K}^a_M(m,m')\,r_a(m')}{\sum_m \sum_{m'} l_a(m)\,c_a(m')\,\widehat{K}^a_M(m,m')\,r_a(m')},$$

or

$$-(\alpha_a)' = \frac{\sum_m l_a(m)\,\varepsilon_a\,\iota_a(m)\;r_a(m)}{\sum_{m'} l_a(m')\,c_a(m')\;r_a(m')}. \tag{3.2.3}$$

Since, $l_a = \pi^a_{\hat{M}}$ and $r_a = 1$, at $\gamma = 0$, thus

$$-(\alpha_a)'(0) = \frac{\sum_m \pi^a_{\hat{M}}(m)\varepsilon_a\,\iota_a(m)}{\sum_{m'} \pi^a_{\hat{M}}(m')\,c_a(m')} = \varepsilon_a\, E_{\pi_a}[A^a_t].$$

The last equality follows from (2.1.1).

3.3 Twisted Process

In this section, we apply a change of measure to twist the sources. Consider an ATM-type source a as in Section 2.1.

For each γ, we constructed a right eigenfunction r_a for the matrix $\widehat{K}^a_M$ with eigenvalue 1, see Section 3.2. In this case, we say that r_a is a harmonic function for the kernel $\widehat{K}^a_M$. We define the twisted probability transition kernel $\hat{\mathcal{K}}^a_\gamma$ as

$$\hat{\mathcal{K}}^a_\gamma(m,m') := \widehat{K}^a_M(m,m')\,\frac{r_a(m')}{r_a(m)}.$$

With the latter kernel, we can generate a process $\hat{\mathcal{M}}^a$. We can view the process as the underlying Markov chain of another ATM-type source. This new source will be called the γ-conjugate of source a.

The underlying Markov chain $\hat{\mathcal{M}}^a$ for the twisted source has the following stationary distribution

$$\varphi^a_{\hat{\mathcal{M}}}(m) := l_a(m)\, r_a(m),$$

since

$$\begin{aligned}\sum_m l_a(m)\, r_a(m)\, \hat{K}^a_\gamma(m,m') &= \sum_m l_a(m)\, \widehat{K}^a_M\, r_a(m') \\ &= l_a(m')\, r_a(m').\end{aligned}$$

We denote $\mathcal{U}^a_t$ as the residual number of time slots until the next state transition of the Markov chain $\hat{\mathcal{M}}^a$. The stationary distribution of the Markov chain $(\mathcal{M}^a, \mathcal{U}^a)$ is denoted as φ_a. The mean number of cells offered by the γ-conjugate source (per time slot) is

$$E_{\varphi_a}[\mathcal{A}^a_t] = \frac{\sum_m \varphi^a_{\hat{\mathcal{M}}}(m)\, \iota_a(m)}{\sum_m \varphi^a_{\hat{\mathcal{M}}}(m)\, c_a(m)}, \tag{3.3.1}$$

where $\mathcal{A}^a_t$ will represent the number of cells offered by the γ-conjugate of source a during time slot t.

Now we show that the derivative of $-\alpha_a$ is equal the mean encumbrance offered per time slot by the γ-conjugate of source a.

Since $\varphi^a_{\hat{\mathcal{M}}}(m) = l_a(m)\, r_a(m)$, then by (3.2.3), we obtain

$$-(\alpha_a)' = \varepsilon_a \frac{\sum_m \varphi^a_{\hat{\mathcal{M}}}(m)\, \iota_a(m)}{\sum_{m'} \varphi^a_{\hat{\mathcal{M}}}(m')\, c_a(m')},$$

which is the desired result.

3.3.1 Alternating On-Off Sources

In this subsection, we consider the special case of the VBR sources of Example 2.1.2. Let source a be such a source.

The operator $\widehat{K}_M^a(m, m'|\alpha_a, \gamma)$ on $\{0,1\} \times \{0,1\}$ in matrix form is given below

$$\begin{bmatrix} e^{\alpha_a} p_{00}^a & e^{(\alpha_a \kappa_a)} p_{01}^a \\ e^{(\alpha_a + \varepsilon_a \gamma)} p_{10}^a & e^{(\alpha_a \kappa_a + \varepsilon_a \gamma)} p_{11}^a \end{bmatrix}.$$

Since $\widehat{K}_M^a$ is a 2×2 matrix with positive entries, it can easily be shown that the Perron-Frobenius eigenvalue for $\widehat{K}_M^a$ is

$$\rho_a(\alpha_a, \gamma) = \frac{1}{2} \left\{ \mathrm{tr}(\widehat{K}_M^a) + \sqrt{\left(\mathrm{tr}(\widehat{K}_M^a)\right)^2 - 4 \det(\widehat{K}_M^a)} \right\}, \tag{3.3.2}$$

where tr and det are the trace and determinant, respectively. In other words,

$$\rho_a(\alpha_a, \gamma) = \frac{e^{\alpha_a}}{2} \left\{ p_{00}^a + p_{11}^a \, e^{(\alpha_a (\kappa_a - 1) + \varepsilon_a \gamma)} + \sqrt{D} \right\},$$

where

$$\begin{aligned} D &:= \left(p_{00}^a + p_{11}^a \, e^{(\alpha_a (\kappa_a - 1) + \varepsilon_a \gamma_A^a)}\right)^2 - 4 \, e^{(\alpha_a (\kappa_a - 1) + \varepsilon_a \gamma_A^a)} \left(p_{00}^a \, p_{11}^a - p_{10}^a \, p_{01}^a\right) \\ &= \left(p_{00}^a - p_{11}^a \, e^{(\alpha_a (\kappa_a - 1) + \varepsilon_a \gamma_A^a)}\right)^2 + 4 \, p_{10}^a \, p_{01}^a \, e^{(\alpha_a (\kappa_a - 1)) + \varepsilon_a \gamma_A^a)} > 0. \end{aligned}$$

For fixed γ, we can solve $\rho_a(\alpha_a, \gamma) = 1$ numerically to obtain $\alpha_a(\gamma)$.

Consider the underlying Markov chain $\hat{\mathcal{M}}^a$ for source a with transition kernel $\hat{\mathcal{K}}_\gamma^a$, see Section 3.3. The probability of remaining *on* is

$$\begin{aligned} \hat{\mathcal{K}}_\gamma^a(1,1) &= \widehat{K}_M^a(1,1) \frac{r_a(1)}{r_a(1)} \\ &= \exp\{\alpha_a(\gamma) \, \kappa_a + \varepsilon_a \gamma\} \, p_{11}^a. \end{aligned}$$

Thus, the probability of the transition from *on* to *off* is

$$\hat{\mathcal{K}}_\gamma^a(1,0) = 1 - \exp\{\alpha_a(\gamma) \, \kappa_a + \varepsilon_a \gamma\} \, p_{11}^a.$$

The probability of remaining *off* is

$$\begin{aligned} \hat{\mathcal{K}}_\gamma^a(0,0) &= \widehat{K}_M^a(0,0) \frac{r_a(0)}{r_a(0)} \\ &= \exp\{\alpha_a(\gamma)\} \, p_{00}^a. \end{aligned}$$

Thus, the probability of the transition from *off* to *on* is

$$\hat{\mathcal{K}}_\gamma^a(1,0) = 1 - \exp\{\alpha_a(\gamma)\} \, p_{00}^a.$$

3.4 Linear Bound and Convexity

In this section, we establish an inequality which will be used in Chapter 4 to prove the existence of a harmonic function. We end with a proof of the convexity of $-\alpha_a$.

Condition 3.4.1 (Non-Trivial Geometric Burst Period) *We will make the assumption that at least one of the burst periods is non-trivial, in the following sense: There exists an on state m such that $\hat{K}^a(m,m) > 0$. In other words, once the process hits the state m it will remain there for a geometric number of transitions.*

We will define the peak cell rate for source a as

$$\mathsf{PCR}_a := \max\left\{\frac{1}{c_a(m)} : m \text{ such that } \hat{K}^a(m,m) > 0, \iota_a(m) = 1\right\}.$$

For both the CBR and the VBR sources of Example 2.1.1 and Example 2.1.2, respectively, the peak cell rate equals the constant inter-arrival rate, i.e. $\mathsf{PCR}_a = \kappa_a$.

Lemma 3.4.2 *The function $-\alpha_a$ is bounded below linearly as follows :*

$$-\alpha_a(\gamma) \geq \varepsilon_a \,\mathsf{PCR}_a \gamma + C,$$

where C is some constant.

Proof : If a is a CBR source then the result is trivial since

$$-\alpha_a(\gamma) = \varepsilon_a \,\kappa_a^{-1}\,\gamma = \varepsilon_a \,\mathsf{PCR}_a \gamma.$$

Now let a be a general ATM-type source and let m be an *on* state such that $1/c_a(m) = \mathsf{PCR}_a$. Since r_a is a harmonic function for $\widehat{K}^a_M$, we obtain

$$\sum_{m'} \widehat{K}^a_M(m,m')\frac{r_a(m')}{r_a(m)} = 1.$$

Since all the terms in the sum are non-negative, this implies that

$$\widehat{K}^a_M(m,m) = \widehat{K}^a_M(m,m)\frac{r_a(m)}{r_a(m)} \leq 1.$$

Thus, by the definition of $\widehat{K}^a_M$,

$$\exp\{\alpha_a\, c_a(m) + \varepsilon_a\, \gamma\, \iota_a(m)\}\ \hat{K}^a(m,m) \leq 1.$$

Since, $1/c_a(m) = \mathsf{PCR}_a$ and $\iota_a(m) = 1$ (since m is an *on* state), then

$$-\alpha_a(\gamma) \geq \varepsilon_a\, \mathsf{PCR}_a \gamma + \mathsf{PCR}_a\, \ln(\hat{K}^a(m,m)).$$

□

Lemma 3.4.3 *The function $\hat{a}^a_\gamma$ defined as*

$$\hat{a}^a_\gamma(m,u) := e^{\alpha_a\, u}\, r_a(m),$$

is a right eigenvector for the following "Feynman-Kac" operator

$$G^a_\gamma(m,u;m',u') := E\left[\exp(\gamma\, \varepsilon_a\, A^a_t)\, \mathbf{1}(M^a_t = m', U^a_t = u') |\, M^a_{t-1} = m, U^a_{t-1} = u\right],$$

with eigenvalue $\exp\{-\alpha_a(\gamma)\}$.

Proof : **Case 1 :** Assume that $u \neq 0$. This implies that with probability one, $M^a_t = m, U^a_t = u-1$, and $A^a_t = 0$. Thus,

$$G^a_\gamma(m,u;m',u') = \begin{cases} 1, & \text{if } m' = m, u' = u-1, \\ 0, & \text{otherwise.} \end{cases}$$

Therefore,

$$\sum_{(m',u')} G^a_\gamma(m,u;m',u')\, e^{\alpha_a\, u'}\, r_a(m') = e^{-\alpha_a}\, e^{\alpha_a\, u}\, r_a(m).$$

Case 2 : Assume that $u = 0$. We will use the fact that r_a is a harmonic function for $\widehat{K}^a_M$ in the following argument :

$$\begin{aligned}
&\sum_{(m',u')} G^a_\gamma(m,u;m',u')\, e^{\alpha_a\, u'}\, r_a(m') \\
=\ &\sum_{(m',u')} \mathbf{1}(u' = c_a(m'))\, e^{[\alpha_a\,(c_a(m')-1)+\varepsilon_a\,\gamma\,\iota_a(m)]}\, \hat{K}^a(m,m') \\
=\ &\sum_{m'} e^{-\alpha_a}\, e^{[\alpha_a\, c_a(m')+\varepsilon_a\,\gamma\,\iota_a(m)]}\, \hat{K}^a(m,m')\, r_a(m') \\
=\ &\sum_{m'} e^{-\alpha_a}\, \widehat{K}^a_M(m,m')\, r_a(m') \\
=\ &e^{-\alpha_a}\, e^{\alpha_a\, u}\, r_a(m) \quad (\text{since } u = 0).
\end{aligned}$$

□

The arguments utilized in the following proof are found in [DO98].

Theorem 3.4.4 *The function $-\alpha_a(\cdot)$ is convex.*

Proof : Consider the matrix G^a_γ from Lemma 3.4.3, with eigenvalue $\exp\{-\alpha_a\}$ and positive right eigenvector $\hat{a}^a_\gamma$.

Define

$$\varepsilon := \max_{(m,u)}\{\hat{a}^a_\gamma(m,u)\} \quad \text{and} \quad \nu := \min_{(m,u)}\{\hat{a}^a_\gamma(m,u)\}.$$

Denote, the kth power of the matrix G^a_γ as $G^{a\,\{k\}}_\gamma$. For all i and j, we have

$$\frac{1}{\varepsilon}\, G^{a\,\{k\}}_\gamma(m,u;m',u')\,\hat{a}^a_\gamma(m',u') \le G^{a\,\{k\}}_\gamma(m,u;m',u') \le \frac{1}{\nu}\, G^{a\,\{k\}}_\gamma(m,u;m',u')\,\hat{a}^a_\gamma(m',u').$$

Sum over (m',u'), take the ln and multiply by $1/k$, we get

$$\begin{aligned}
&\frac{1}{k}\ln\left\{\sum_{(m',u')} G^{a\,\{k\}}_\gamma(m,u;m',u')\,\hat{a}^a_\gamma(m',u')\right\} - \frac{1}{k}\ln\varepsilon \\
\le &\frac{1}{k}\ln\left\{\sum_{(m',u')} G^{a\,\{k\}}_\gamma(m,u;m',u')\right\} \\
\le &\frac{1}{k}\ln\left\{\sum_{(m',u')} G^{a\,\{k\}}_\gamma(m,u;m',u')\,\hat{a}^a_\gamma(m',u')\right\} - \frac{1}{k}\ln\nu.
\end{aligned}$$

Hence,

$$\lim_{k\to\infty}\frac{1}{k}\ln\left\{\sum_{(m',u')} G^{a\,\{k\}}_\gamma(m,u;m',u')\right\} = -\alpha_a, \tag{3.4.1}$$

since $G^{a\,\{k\}}_\gamma\,\hat{a}^a_\gamma = \left(\exp\{-\alpha_a\}\right)^k\,\hat{a}^a_\gamma$.

Define

$$M^a_A(\gamma,k) := E_{(m,u)}\left[\exp\left(\gamma\sum_{t=1}^{k}\varepsilon_a\,A^a_t\right)\right] = \sum_{(m',u')} G^{a\,\{k\}}_\gamma(m,u;m',u'),$$

where A_t^a is one if source a offers a cell during the time slot t, otherwise it is zero. The functions in the sequence $\{\ln M_A^a(\gamma, n)\}_n$ are all convex functions of γ, since logarithms of moment generating functions are convex functions. Thus,

$$\lim_{k\to\infty} \frac{1}{k} \ln\left(M_A^a(\gamma, k)\right) = -\alpha_a(\gamma)$$

is also a convex function since convexity is closed under limits.

□

3.5 Phase Shift

In Chapter 4, we will apply a phase shift to $-\alpha_a$ in order to construct a Lyapounov function. We investigate the consequences of this shift.

Let θ be some positive constant. We consider the θ-conjugate of source a, its mean arrival rate is $E_{\varphi_a}[A_t^a] = -(\alpha_a)'(\theta)/\varepsilon_a$.

Lemma 3.5.1 *The function $\hat{a}_s^{a*}$ defined as $\hat{a}_s^{a*}(m,u) := \hat{a}_\gamma^a(m,u)$, at $\gamma = \theta - s/\varepsilon_a$, is a right eigenvector for the matrix defined as follows*

$$G_s^{a*}(m,u;m',u') = E\left[\exp((\theta\,\varepsilon_a - s)\,A_t^a)\,\mathbf{1}(M_t^a = m', U_t^a = u') \middle|\, M_{t-1}^a = m, U_{t-1}^a = u\right],$$

with eigenvalue $\exp\{-\alpha_a^*(s)\}$, where $\alpha_a^*(s) := \alpha_a(\theta - s/\varepsilon_a)$.

Furthermore, $-(\alpha_a^*)'(0) = -E_{\varphi_a}[A_t^a]$.

Proof: We start by showing that $G_s^{a*}\,\hat{a}_s^{a*} = e^{-\alpha_a^*(s)}\,\hat{a}_s^{a*}$. Since $\theta\,\varepsilon_a - s = (\theta - s/\varepsilon_a)\,\varepsilon_a$, then $G_s^{a*} = G_\gamma^a$, at $\gamma = \theta - s/\varepsilon_a$. The result then follows as a consequence of Lemma 3.4.3, i.e. that $\hat{a}_\gamma^a$ is a right eigenvector for G_γ^a with eigenvalue $\exp(-\alpha_a(\gamma))$.

Now, we differentiate $-\alpha_a^*$ with respect to s. We obtain

$$\frac{d}{ds}(-\alpha_a(\theta - s/\varepsilon_a)) = -(\alpha_a)'(\theta - s/\varepsilon_a) \times (-1/\varepsilon_a).$$

Thus,

$$-(\alpha_a^*)'(0) = -(\alpha_a)'(\theta) \times (-1/\varepsilon_a) = -E_{\varphi_a}[A_t^a],$$

since $E_{\varphi_a}[A_t^a] = -(\alpha_a)'(\theta)/\varepsilon_a$.

□

3.6 The Nodes

For each tagged node $b \in TN$, we define the generating function of its encumbrance service process as

$$\mathcal{M}_b(\beta_b, \gamma) := \exp\{\beta_b\, d_b + d_b\, \gamma\}.$$

Recall, the encumbrance due to node b is d_b and that for every node b, the total encumbrance is reduced by d_b every d_b time slots.

For $\gamma \in \mathbb{R}$, we will define $\beta_b(\gamma)$ as the unique β_b that satisfies $\mathcal{M}_b(\beta_b, \gamma) = 1$. Hence,

$$\beta_b(\gamma) = -\gamma.$$

It follows directly that $-\beta_b(-\gamma) = -\gamma$ is a convex and differentiable function in γ, $\beta_b(0) = 0$, and

$$\frac{d}{d\gamma}\{-\beta_b(-\gamma)\} = -1, \quad \text{for all } \gamma \in \mathbb{R}.$$

Lemma 3.6.1 *The function* $\hat{a}^b_\gamma$ *defined as*

$$\hat{a}^b_\gamma(v) := e^{\beta_b\, v},$$

is a right eigenvector for the following "Feynman-Kac" operator

$$H^b_\gamma(v; v') := E\left[\exp(\gamma\, d_b\, B^b_t)\, \mathbf{1}(V^b_t = v') \middle| V^b_{t-1} = v\right],$$

with eigenvalue $\exp\{-\beta_b(\gamma)\}$*, where* B^b_t *is one or zero if time slot* t *is a service slot for node* b*.*

Proof : **Case 1 :** Assume that $v \neq 0$. This implies that with probability one, $V^b_t = v - 1$, and $B^b_t = 0$. Thus,

$$H^b_\gamma(v, v') = \begin{cases} 1, & \text{if } v' = v - 1, \\ 0, & \text{otherwise.} \end{cases}$$

Therefore,

$$\sum_{v'} H^b_\gamma(v; v')\, e^{\beta_b\, v'} = e^{-\beta_b}\, e^{\beta_b\, v}.$$

Case 2 : Assume that $v = 0$. This implies that with probability one, $V_t^b = d_b - 1$, and $B_t^b = 1$. Thus,

$$H_\gamma^b(v, v') = \begin{cases} e^{\gamma\, d_b}, & \text{if } v' = d_b - 1, \\ 0, & \text{otherwise.} \end{cases}$$

Therefore,

$$\sum_{v'} H_\gamma^b(v; v')\, e^{\beta_b\, v'} = e^{\gamma\, d_b}\, e^{\beta_b\,(d_b - 1)} = e^{-\beta_b} = e^{-\beta_b}\, e^{\beta_b\, v},$$

since $v = 0$.

□

We finish with a phase shift result.

Lemma 3.6.2 *Let θ be some positive real number. The function $\hat{a}_s^{b*}$ defined as $\hat{a}_s^{b*}(v) := \hat{a}_\gamma^b(v)$ at $\gamma = s/d_b - \theta$, is a right eigenvector for the following operator*

$$H_s^{b*}(v; v') := E\left[\exp((s - \theta\, d_b)\, B_t^a)\, \mathbf{1}(V_t^b = v') \middle| V_{t-1}^b = v\right],$$

with eigenvalue $\exp\{-\beta_b^*(s)\}$, where $\beta_b^*(s) := \beta_b(s/d_b - \theta) = -(s/d_b - \theta)$.

Furthermore,

$$\frac{d}{ds}\left(-\beta_b^*(s)\right) = \frac{1}{d_b}.$$

Proof : Since $s - \theta\, d_b = (s/d_b - \theta)\, d_b$, then $H_s^{b*} = H_\gamma^b$ with $\gamma = s/d_b - \theta$. The result then follows from Lemma (3.6.1).

Now we differentiate,

$$\frac{d}{ds}\left(-\beta_b^*(s)\right) = \frac{d}{ds}\left(s/d_b - \theta)\right) = \frac{1}{d_b}.$$

□

Chapter 4

Large Deviations

4.1 Introduction

This section serves as a brief introduction to the large deviation theory found in [McDo99, FM03]. In the general setting of the latter references, we have a Markov chain W, with transition kernel K, stationary measure π and state space $S \subseteq \mathbb{Z}_+ \times \mathbb{Z}^{r-1} \times \hat{S}$, where $\hat{S}$ is a countable state space. In our case, $\hat{S}$ is finite. The state space is extended to $S^\infty \equiv \mathbb{Z}^r \times \hat{S}$, in order to construct a Markov additive process $W^\infty = (\tilde{W}^\infty, \hat{W}^\infty)$, on S^∞, where $\tilde{W}^\infty$ is the additive part and $\hat{W}^\infty$ is the Markovian part (sometimes called the "environment"). The transition kernel for W^∞ is denoted K^∞. The process W^∞ is called the free process. It should be noted that the Markov additive condition on W^∞ can be weakened, see [FM03], or the end of Section 1.1 in [McDo99]. They only consider the weaker condition that the marginal process $(\tilde{W}_1^\infty, \hat{W}^\infty)$ is a Markov additive process, where $\tilde{W}_1^\infty$ is the first component of the process $\tilde{W}^\infty$. We will use $\mathcal{J}$ to denote the transition kernel of the Markov additive chain $(\tilde{W}_1^\infty, \hat{W}^\infty)$.

It is assumed that there exists a boundary $\blacktriangle \subset S^\infty$ such that the transition kernels for W and W^∞ agree within the interior $S^\circ = S^\infty \setminus \blacktriangle$ of S, i.e.

$$K(w, w') = K^\infty(w, w'), \quad \text{if } w \in S^\circ \text{ and } w' \in S^\circ.$$

The edge of the boundary $\blacktriangle$ is denoted $\Delta \equiv \blacktriangle \cap S$. It is also assumed that there is

a positive harmonic function h for the transition kernel K^∞, that is h satisfies the following averaging property

$$h(w) = \sum_{w' \in S^\infty} K^\infty(w, w')\, h(w').$$

Furthermore, it is of the form

$$h(\tilde{w}, \hat{w}) = \exp(\theta\, \tilde{w}_1)\, \hat{a}(\hat{w}),$$

where $\tilde{w}_1$ is the first component of the vector $\tilde{w}$, θ is a positive real constant and $\hat{a}(\cdot)$ is a positive real function on $\hat{S}$.

The change of measure or *twist* is constructed using the harmonic function h. The h-transform for K^∞ is

$$\mathcal{K}^\infty(w, w') := K^\infty(w, w')h(w')/h(w).$$

We now present some technical conditions similar to conditions found in [McDo99]. Note, the conditions from [McDo99] which are not mentioned below are trivial since the state space $\hat{S}$ of the Markovian part is finite. Following these conditions, we present a large deviation result which follows directly from Theorem 1.6 in [McDo99].

C.1: The marginal Markov chain $\{\hat{\mathcal{W}}^\infty[t], t \in \mathbb{Z}_+\}$, with transition kernel $\hat{\mathcal{K}}^\infty$ has a stationary probability distribution $\varphi(\cdot)$.

C.2: Let T_n be the sequence of return times for the Markovian part $\hat{\mathcal{W}}^\infty$ to its initial state. We will assume that the process $\{\tilde{\mathcal{W}}_1^\infty[T_n] : n \in \mathbb{Z}_+\}$ is aperiodic.

C.3: The first coordinate of the drift vector of the stationary version of $\mathcal{W}^\infty$ has a finite, strictly positive drift. That is, $0 < \tilde{d}_1 < \infty$ where

$$\tilde{d} = \sum_{\hat{w} \in \hat{S}} \varphi(\hat{w})\, E\left[\tilde{\mathcal{W}}^\infty[1] \,\middle|\, \mathcal{W}^\infty[0] = (0, \hat{w})\right].$$

C.4: The twisted free process starting from Δ has a positive probability of never hitting $\blacktriangle$, i.e.

$$\sum_{w \in \Delta} \pi(w)\, P_w(\mathcal{T}_\blacktriangle^\infty = \infty) > 0,$$

where $\mathcal{T}_\blacktriangle^\infty \equiv \{n > 0 | \mathcal{W}^\infty[n] \in \blacktriangle\}$.

C.5: Define the function $f(w) := h(w)\,1(w \in \Delta)$. We need

$$\pi(f) := \sum_{w \in S} \pi(w)\, f(w) < \infty.$$

Theorem 4.1.1 *Under C.1-C.5 and as $\ell \to \infty$, for $w \in S$ such that $\tilde{w}_1 = \ell$, then*

$$\pi(w) \sim C_{\hat{w}} \exp(-\theta\, \ell),$$

where $C_{\hat{w}} \in \mathbb{R}_+$ for all $\hat{w} \in \hat{S}$.

In Section 4.2, we construct the harmonic function h and give sufficient conditions for its existence. We also see that provided that h does exist the marginal Markov chain $\{\hat{W}^\infty[t], t \in \mathbb{Z}_+\}$, with transition kernel $\hat{\mathcal{K}}^\infty$ has a stationary probability distribution $\varphi(\cdot)$. Furthermore, under the same sufficient conditions as given for the existence of h, C.3 will follow.

In Section 4.3, we show that Condition C.5 is a sufficient condition for Condition 6 in [McDo99]. In Section 4.5, we investigate C.4 and C.5 for a simplified model, i.e. a two node feedforward network of queues in tandem.

4.2 h-Transform

At the end of Section 2.5, we defined the process $W^\infty = (\tilde{W}^\infty, \hat{W}^\infty)$ as

$$\tilde{W}^\infty[t] = \left(ENC_t^\infty, (\mathfrak{c}^b{}_t)_{b \in B}\right) \quad \text{and} \quad \hat{W}^\infty[t] = \left((V^b{}_t)_{b \in B}, (M^a{}_t, U^a{}_t)_{a \in A}\right).$$

We also remarked that the marginal process $(\tilde{W}_1^\infty[t], \hat{W}^\infty[t])$ is a Markov additive process with transition kernel $\mathcal{J}$.

The marginal process $(\tilde{W}_1^\infty[t], \hat{W}^\infty[t])$ is a special case of the free process found in [BDM99]. We will use their construction of the harmonic function h for K^∞.

4.2.1 Construction of h

In this section, we construct a harmonic function h, for the transition kernel K^∞, of the form

$$h(\tilde{w}, \hat{w}) = \exp(\theta\, \tilde{w}_1)\, \hat{a}(\hat{w}),$$

where θ is a positive real constant, $\hat{a}$ is a positive real-valued function on $\hat{S}$. It must satisfy the following property

$$\sum_{w' \in S^{\infty}} K^{\infty}(w; w')\, h(w') = h(w).$$

Since h only depends on the first component of $\tilde{w}$, then it is sufficient to show that h restricted to the domain $\mathbb{Z} \times \hat{S}$, also denoted h, is a harmonic function for the marginal process

$$(\tilde{W}_1^{\infty}[t], \hat{W}^{\infty}[t]) = \left(ENC_t^{\infty}, \left((V^b{}_t)_{b \in B}, (M^a{}_t, U^a{}_t)_{a \in A}\right)\right),$$

with transition kernel $\mathcal{J}$. We will show that

$$\sum_{(e', \vec{v}', \vec{m}', \vec{u}') \in \mathbb{Z} \times \hat{S}} \mathcal{J}(e, \vec{v}, \vec{m}, \vec{u}; e', \vec{v}', \vec{m}', \vec{u}')\, h(e', \vec{v}', \vec{m}', \vec{u}') = h(e, \vec{v}, \vec{m}, \vec{u}).$$

Let $\gamma \in \mathbb{R}$, respectively define $\hat{a}_\gamma$ on $\hat{S}$ and h_γ on $\mathbb{Z} \times \hat{S}$ as

$$\hat{a}_\gamma(\vec{v}, \vec{m}, \vec{u}) := \prod_{a \in T_{vc} \cup C_{vc}} \hat{a}_\gamma^a(m_a, u_a) \times \prod_{b \in TN} \hat{a}_{-\gamma}^b(v_b),$$

where $\hat{a}_\gamma^a(m_a, u_a)$ and $\hat{a}_\gamma^b(v_b)$ are taken from Lemma 3.4.3 and Lemma 3.6.1, respectively, and

$$h_\gamma(e, \vec{v}, \vec{m}, \vec{u}) := \exp(\gamma\, e)\, \hat{a}_\gamma(\vec{v}, \vec{m}, \vec{u}). \tag{4.2.1}$$

Recall that the total encumbrance ENC^{∞}, for the free process, satisfies the recurrence given by (2.5.3). This is equivalent to

$$ENC_t^{\infty} - ENC_{t-1}^{\infty} = \sum_{a \in T_{vc} \cup C_{vc}} \varepsilon_a\, A_t^a - \sum_{b \in TN} d_b\, B_t^b,$$

where A_t^a, B_t^b are indicator variables for the cell arrival from source a and cell served at node b, respectively. Furthermore, as noted at the end of Chapter 3, that for fixed $\hat{w}$, the difference $\varsigma(e, \hat{w}) - e$ is a constant and that conditioned on $\hat{W}[t-1] = \hat{w}$, we have

$$\sum_{a \in T_{vc} \cup C_{vc}} \varepsilon_a\, A_t^a - \sum_{b \in TN} d_b\, B_t^b = \varsigma(e, \hat{w}) - e,$$

almost surely. Furthermore, the form of the kernel $\mathcal{J}$ is given by (2.5.4).

Consider the following sum,

$$\begin{aligned}
&\sum_{(e',\vec{v}',\vec{m}',\vec{u}')\in\mathbb{Z}\times\hat{S}} \mathcal{J}(e,\vec{v},\vec{m},\vec{u};e',\vec{v}',\vec{m}',\vec{u}')\, e^{-\gamma e}\, h_\gamma(e',\vec{v}',\vec{m}',\vec{u}') \\
&= \sum_{(e',\vec{v}',\vec{m}',\vec{u}')\in\mathbb{Z}\times\hat{S}} \mathcal{J}(e,\vec{v},\vec{m},\vec{u};e',\vec{v}',\vec{m}',\vec{u}')\, e^{\gamma(e'-e)}\, \hat{a}_\gamma(\vec{v}',\vec{m}',\vec{u}') \\
&= \sum_{\hat{w}'\in\hat{S}} e^{\gamma(\varsigma(e,\hat{w})-e)}\, \mathsf{P}\left[\hat{W}^\infty[1]=\hat{w}'|\hat{W}^\infty[0]=\hat{w}\right]\, \hat{a}_\gamma(\hat{w}'), \qquad (4.2.2)
\end{aligned}$$

where $\hat{w}=(\vec{v},\vec{m},\vec{u})$ and $\hat{w}'=(\vec{v}',\vec{m}',\vec{u}')$. The last equality follows from the fact that the kernel $\mathcal{J}$ is of the form

$$\mathcal{J}(e,\hat{w};e',\hat{w}') = \mathbf{1}(e'=\varsigma(e,\hat{w}))\,\mathsf{P}\left[\hat{W}^\infty[1]=\hat{w}'|\hat{W}^\infty[0]=\hat{w}\right].$$

The sum (4.2.2), is equal to

$$\begin{aligned}
&\sum_{\hat{w}'\in\hat{S}} E_{\hat{w}}\left[e^{\gamma(\varsigma(e,\hat{w})-e)}\,\mathbf{1}\left(\hat{W}^\infty[1]=\hat{w}'\right)\right]\,\hat{a}_\gamma(\hat{w}') \\
&= \sum_{\hat{w}'\in\hat{S}} E_{\hat{w}}\left[e^{\gamma\left(\sum_{a\in T_{vc}\cup C_{vc}}\varepsilon_a A_t^a-\sum_{b\in TN} d_b B_t^b\right)}\,\mathbf{1}\left(\hat{W}^\infty[1]=\hat{w}'\right)\right]\,\hat{a}_\gamma(\hat{w}')
\end{aligned}$$

From the independence of the sources and the nodes, we obtain the following,

$$\begin{aligned}
&\prod_{a\in T_{vc}\cup C_{vc}}\sum_{(m_a',u_a')} E_{(m_a,u_a)}\left[\exp(\gamma\,\varepsilon_a\,A_1^a)\,\mathbf{1}(M_1^a=m',U_1^a=u')\right]\,\hat{a}_\gamma^a(m_a',u_a') \\
&\times\prod_{b\in TN}\sum_{v_b'} E_{v_b}\left[\exp(-\gamma\,d_b\,B_1^b)\,\mathbf{1}(V_1^b=v')\right]\,\hat{a}_\gamma^b(v_b') \\
&= \prod_{a\in T_{vc}}\exp\{\alpha_a(\gamma)\}\,\hat{a}_\gamma^a(m_a,u_a)\times\prod_{b\in TN}\exp\{\beta_b(\gamma)\}\,\hat{a}_{-\gamma}^b(v_b) \qquad (4.2.3) \\
&= \exp\left\{\sum_{a\in T_{vc}\cup C_{vc}}-\alpha_a(\gamma)+\sum_{b\in TN}-\beta_b(-\gamma)\right\}\prod_{a\in T_{vc}}\hat{a}_\gamma^a(m_a,u_a)\times\prod_{b\in TN}\hat{a}_{-\gamma}^b(v_b), \\
&= \exp\left\{\sum_{a\in T_{vc}\cup C_{vc}}-\alpha_a(\gamma)+\sum_{b\in TN}-\beta_b(-\gamma)\right\}\hat{a}_\gamma(\vec{v},\vec{m},\vec{u})
\end{aligned}$$

where (4.2.3) follows from Lemma 3.4.3 and Lemma 3.6.1.

Define

$$\Lambda(\gamma) := \sum_{a \in T_{vc} \cup C_{vc}} -\alpha_a(\gamma) + \sum_{b \in TN} -\beta_b(-\gamma).$$

We have shown that

$$\sum_{(e',\vec{v}',\vec{m}',\vec{u}') \in \mathbb{Z} \times \hat{S}} \mathcal{J}(e,\vec{v},\vec{m},\vec{u};e',\vec{v}',\vec{m}',\vec{u}')\, e^{-\gamma e}\, h_\gamma(e',\vec{v}',\vec{m}',\vec{u}') = \exp\{\Lambda(\gamma)\}\, \hat{a}_\gamma(\vec{v},\vec{m},\vec{u}).$$

Multiplying both sides by $e^{\gamma e}$, we obtain

$$\sum_{(e',\vec{v}',\vec{m}',\vec{u}') \in \mathbb{Z} \times \hat{S}} \mathcal{J}(e,\vec{v},\vec{m},\vec{u};e',\vec{v}',\vec{m}',\vec{u}')\, h_\gamma(e',\vec{v}',\vec{m}',\vec{u}') = \exp\{\Lambda(\gamma)\}\, h_\gamma(e,\vec{v},\vec{m},\vec{u}).$$

Therefore, h_γ is a harmonic function for $\mathcal{J}$, if $\Lambda(\gamma) = 0$. The result is stated in the following theorem.

Theorem 4.2.1 *Assume that there exists a positive real number θ such that*

$$-\sum_{a \in T_{vc} \cup C_{vc}} \alpha_a(\theta) - \sum_{b \in TN} \beta_b(-\theta) = 0,$$

that is $\Lambda(\theta) = 0$. The function h of $\mathbb{Z} \times \hat{S}$ defined as $h := h_\gamma$, with $\gamma = \theta$, is a harmonic function for the transition kernel $\mathcal{J}$.

4.2.2 Twisted Process

In the prequel, we constructed a harmonic function h for K^∞. Using h, we construct the h-transform for K^∞, denoted $\mathcal{K}^\infty$, as

$$\mathcal{K}^\infty(w;w') := K^\infty(w;w')\frac{h(w')}{h(w)}.$$

The latter is a probability transition kernel. The process generated by it will be denoted $\mathcal{W}^\infty$.

Assume that w and w' are two states such that $K^\infty(w;w') > 0$, then

$$K^\infty(w;w') = \prod_{a:u_a=0} \hat{K}^a(m_a, m'_a),$$

since all of the randomness is in the sources. The ratio $h(w')/h(w)$ is equal to

$$
\begin{aligned}
& e^{\theta(e'-e)} \frac{\prod_{a\in T_{vc}\cup C_{vc}} \hat{a}^a_\theta(m'_a,u'_a) \times \prod_{b\in TN} \hat{a}_{-\theta}(v'_b)}{\prod_{a\in T_{vc}\cup C_{vc}} \hat{a}^a_\theta(m_a,u_a) \times \prod_{b\in TN} \hat{a}_{-\theta}(v_b)} \\
= \; & e^{\theta(e'-e)} \frac{\prod_{a\in T_{vc}\cup C_{vc}} e^{\alpha_a(\theta)\,u'_a}\, r_a(m'_a) \times \prod_{b\in TN} e^{\beta_b(-\theta)\,v'_b}}{\prod_{a\in T_{vc}\cup C_{vc}} e^{\alpha_a(\theta)\,u_a}\, r_a(m_a) \times \prod_{b\in TN} e^{\beta_b(-\theta)\,v_b}} \\
= \; & e^{\theta(e'-e)} \prod_{a\in T_{vc}\cup C_{vc}} e^{\alpha_a(\theta)(u'_a-u_a)} \frac{r_a(m'_a)}{r_a(m_a)} \times \prod_{b\in TN} e^{\beta_b(-\theta)\,(v'_b-v_b)} \\
= \; & e^{\theta(e'-e)} \prod_{a\in T_{vc}\cup C_{vc}\,:\,u_a=0} e^{\alpha_a(\theta)(c_a(m'_a)-1)} \frac{r_a(m'_a)}{r_a(m_a)} \times \prod_{a\in T_{vc}\cup C_{vc}\,:\,u_a\neq 0} e^{-\alpha_a(\theta)} \\
& \times \prod_{b\in TN\,:\,v_b=0} e^{\beta_b(-\theta)\,(d_b-1)} \times \prod_{b\in TN\,:\,v_b\neq 0} e^{-\beta_b(-\theta)} \\
= \; & \prod_{a\in T_{vc}\cup C_{vc}\,:\,u_a=0} e^{\theta\,\varepsilon_a\,\iota(m_a)}\, e^{\alpha_a(\theta)(c_a(m'_a)-1)} \frac{r_a(m'_a)}{r_a(m_a)} \times \prod_{a\in T_{vc}\cup C_{vc}\,:\,u_a\neq 0} e^{-\alpha_a(\theta)} \\
& \times \prod_{b\in TN\,:\,v_b=0} e^{-\theta\,d_b} e^{\beta_b(-\theta)\,(d_b-1)} \times \prod_{b\in TN\,:\,v_b\neq 0} e^{-\beta_b(-\theta)}.
\end{aligned}
$$

Since

$$\sum_{a\in T_{vc}\cup C_{vc}} -\alpha_a(\theta) - \sum_{b\in TN} \beta(-\theta) = 0,$$

and that $\beta_b(-\theta) = \theta$, then

$$\frac{h(w')}{h(w)} = \prod_{a\in T_{vc}\cup C_{vc}\,:\,u_a=0} e^{\theta\,\varepsilon_a\,\iota(m_a)}\, e^{\alpha_a(\theta)\,c_a(m'_a)} \frac{r_a(m'_a)}{r_a(m_a)}.$$

Thus, the h-transform $\mathcal{K}^\infty(w,w')$, is equal to

$$
\begin{aligned}
& \prod_{a:u_a=0} \hat{K}^a(m_a,m'_a) \times \prod_{a\in T_{vc}\cup C_{vc}\,:\,u_a=0} e^{\theta\,\varepsilon_a\,\iota(m_a)}\, e^{\alpha_a(\theta)\,c_a(m'_a)} \frac{r_a(m'_a)}{r_a(m_a)} \\
= \; & \prod_{a\in R_{vc}\,:\,u_a=0} \hat{K}^a(m_a,m'_a) \times \\
& \prod_{a\in T_{vc}\cup C_{vc}\,:\,u_a=0} e^{\theta\,\varepsilon_a\,\iota(m_a)}\, e^{\alpha_a(\theta)\,c_a(m'_a)}\, \hat{K}^a(m_a,m'_a) \frac{r_a(m'_a)}{r_a(m_a)} \\
= \; & \prod_{a\in R_{vc}\,:\,u_a=0} \hat{K}^a(m_a,m'_a) \times \prod_{a\in T_{vc}\cup C_{vc}\,:\,u_a=0} \widehat{K}^a_M(m_a,m'_a) \frac{r_a(m'_a)}{r_a(m_a)} \\
= \; & \prod_{a\in R_{vc}\,:\,u_a=0} \hat{K}^a(m_a,m'_a) \times \prod_{a\in T_{vc}\cup C_{vc}\,:\,u_a=0} \hat{\mathcal{K}}^a_\theta(m_a,m'_a),
\end{aligned}
$$

where $\hat{\mathcal{K}}_\theta^a$ is equal to $\hat{\mathcal{K}}_\gamma^a$ with $\gamma = \theta$, see Section 3.3.

Thus, for the twisted regime, the transition kernels of the underlying Markov chains of the tagged sources (T_{vc}) or the sources in the cross traffic (C_{vc}) are affected. Everything else remains unchanged.

The marginal process $\hat{\mathcal{W}}^\infty$ with transition kernel $\hat{\mathcal{K}}$ is an irreducible, aperiodic Markov chain. Its stationary distribution is

$$\varphi(\vec{v}, \vec{m}, \vec{u}) = \prod_b \frac{1}{d_b} \times \prod_{a \in R_{vc}} \pi_a(m_a, u_a) \times \prod_{a \in T_{vc} \cup C_{vc}} \varphi_a(m_a, u_a),$$

where φ_a is given in Section 3.3. Therefore, Condition C.1 is satisfied.

4.2.3 Existence

In this section, we provide a sufficient condition for the existence of a positive real number θ that satisfies $\Lambda(\theta) = 0$ where

$$\Lambda(\gamma) := -\sum_{a \in T_{vc} \cup C_{vc}} \alpha_a(\gamma) - \sum_{b \in TN} \beta_b(-\gamma). \tag{4.2.4}$$

Thus, we provide a sufficient condition for the existence of the harmonic function h from Theorem 4.2.1. We present a few lemmas before the main result.

Lemma 4.2.2 *The function $\Lambda(\cdot)$ from (4.2.4) satisfies the following properties*

1. *Λ is convex;*
2. *$\Lambda(0) = 0$.*

Proof :

1. Since the functions $-\alpha_a(\cdot)$, for all a, and $-\beta_b(\cdot)$, for all b, are convex, then the function Λ is convex.
2. Since $-\alpha_a(0) = 0$, for all a, and $-\beta_b(0) = 0$, for all b, then $\Lambda(0) = 0$.

□

Lemma 4.2.3 *If*

$$\sum_{a \in T_{vc} \cup C_{vc}} \varepsilon_a E_{\pi_a}[A_t^a] < |TN|,$$

where $|TN|$ is the number of tagged nodes, then $\Lambda'(0) < 0$.

Proof : We have shown that

$$\frac{d}{d\gamma}(-\alpha_a(\gamma))\Big|_{\gamma=0} = \varepsilon_a \, E_{\pi_a}[A_t^a], \quad \text{for all } a \in T_{vc} \cup C_{vc},$$

and

$$\frac{d}{d\gamma}(-\beta_b(-\gamma))\Big|_{\gamma=0} = -1, \quad \text{for all } b \in TN.$$

Thus, it follows directly that Λ as defined by (4.2.4) has a negative derivative at $\gamma = 0$, i.e. $\Lambda'(0) < 0$.

□

Lemma 4.2.4 *If*

$$\sum_{a \in \in T_{vc} \cup C_{vc}} \varepsilon_a \, \mathsf{PCR}_a - |TN| > 0,$$

where $|TN|$ is the number of tagged nodes, then

$$\lim_{\gamma \to +\infty} \Lambda(\gamma) = +\infty.$$

Proof : By Lemma 3.4.2, for all $a \in T_{vc} \cup C_{vc}$, there exists a constant C_a such that

$$-\alpha_a(\gamma) \geq \varepsilon_a \, \mathsf{PCR}_A \, \gamma + C_a.$$

We have also shown that

$$-\beta_b(-\gamma) = -\gamma, \quad \text{for all } b \in TN.$$

Thus,

$$\Lambda(\gamma) \geq \gamma \left(\sum_{a \in T_{vc} \cup C_{vc}} \varepsilon_a \, \mathsf{PCR}_a - |TN| \right) + \sum_{a \in \in T_{vc} \cup C_{vc}} C_a.$$

The statement of the lemma follows directly from the last inequality.

□

Theorem 4.2.5 *If*

$$\sum_{a\in T_{vc}\cup C_{vc}} \varepsilon_a E_{\pi_a}[A_t^a] < |TN| \quad \text{and} \quad \sum_{a\in T_{vc}\cup C_{vc}} \varepsilon_a \,\mathsf{PCR}_a - |TN| > 0,$$

where $|TN|$ *is the number of tagged nodes, then there exists a positive real number* θ *that satisfies* $\Lambda(\theta) = 0$. *Furthermore, the harmonic function* h *from Theorem 4.2.1 exists.*

Proof : From Lemma 4.2.2, Lemma 4.2.3 and Lemma 4.2.4, the function Λ is convex, $\Lambda(0) = 0$, $\Lambda'(0) < 0$ and $\lim_{\gamma\to+\infty} \Lambda(\gamma) = +\infty$, therefore there exists a unique positive real number θ such that $\Lambda(\theta) = 0$.

□

We end the section with interpretations of the sufficient conditions in the statement of Theorem 4.2.5. We can better understand these conditions by looking at the increment of the total encumbrance of the free process,

$$\sum_{a\in T_{vc}\cup C_{vc}} \varepsilon_a \, A_t^a - \sum_{b\in TN} d_b B_t^b.$$

Hence, the mean drift of the total encumbrance of the free process is

$$\sum_{a\in T_{vc}\cup C_{vc}} \varepsilon_a \, E_{\pi_a}[A_t^a] - |TN|,$$

or under the twist it becomes

$$\sum_{a\in T_{vc}\cup C_{vc}} \varepsilon_a \, E_{\varphi_a}[A_t^a] - |TN|.$$

The first condition, i.e.

$$\sum_{a\in T_{vc}\cup C_{vc}} \varepsilon_a E_{\pi_a}[A_t^a] < |TN|$$

means that we would like the mean drift of the total encumbrance of the free process to be negative. We show that this condition is satisfied if for each tagged node b, the

mean number of cells offered to the system with the tagged node b in their path is less than the mean number of service slots at the tagged node b. In other words,

$$\sum_{a \in A(b)} E_{\pi_a}[A_t^a] < \frac{1}{d_b}, \text{ for all } b \in TN, \quad \text{implies} \quad \sum_{a \in T_{vc} \cup C_{vc}} \varepsilon_a E_{\pi_a}[A_t^a] < |TN|,$$

where $A(b)$ is the index set for the sources that offer cells with node b in their path. By definition, the encumbrance offered by a cell as it enters the network is

$$\varepsilon_a = \sum_{b \in TN} d_b \, \mathbf{1}\{a \in A(b)\}.$$

Hence,

$$\begin{aligned} \sum_{a \in T_{vc} \cup C_{vc}} \varepsilon_a E_{\pi_a}[A_t^a] - |TN| &= \sum_{a \in T_{vc} \cup C_{vc}} \varepsilon_a \, E_{\pi_a}[A_t^a] - |TN| \\ &= \sum_{a \in T_{vc} \cup C_{vc}} E_{\pi_a}[A_t^a] \sum_{b \in TN} d_b \, \mathbf{1}\{a \in A(b)\} - |TN| \\ &= \sum_{b \in TN} \sum_{a \in A(b)} d_b \, E_{\pi_a}[A_t^a] - |TN|. \end{aligned}$$

Since

$$\sum_{a \in A(b)} E_{\pi_a}[A_t^a] < \frac{1}{d_b} \quad \text{for all } b \in TN,$$

then

$$\sum_{a \in A(b)} d_b \, E_{\pi_a}[A_t^a] < 1 \quad \text{for all } b \in TN.$$

Therefore,

$$\sum_{b \in TN} \sum_{a \in A(b)} d_b \, E_{\pi_a}[A_t^a] - |TN| < \sum_{b \in TN} 1 - |TN| = 0,$$

which is the desired result.

The second condition is

$$\sum_{a \in T_{vc} \cup C_{vc}} \varepsilon_a \, \mathsf{PCR}_a - |TN| > 0.$$

If the sources were always *on* and offering cells at their respective peak cell rate, then then left-hand-side of the latter inequality would represent the mean drift for the total

encumbrance of the free process. Thus, this condition states that the largest possible mean drift for the total encumbrance of the free process must be strictly positive. In other words, there exists a configuration of the sources such that the encumbrance of the free process will want to increase on "average".

4.2.4 The Positive Drift

In this subsection, we see that under the conditions of Theorem 4.2.5, Condition C.3 is satisfied. Condition C.3 states that the first coordinate of the drift vector of the stationary version of $\mathcal{W}^\infty$ has a finite, strictly positive drift. That is, $0 < \tilde{d}_1 < \infty$ where

$$\tilde{d} := \sum_{\hat{w} \in \hat{S}} \varphi(\hat{w})\, E\left[\tilde{\mathcal{W}}^\infty[1] \,\middle|\, \mathcal{W}^\infty[0] = (0, \hat{w}) \right].$$

As defined in the latter condition, $\tilde{d}_1$ represents the mean increment for the total encumbrance under the twist, i.e.

$$\tilde{d}_1 = \sum_{a \in T_{vc} \cup C_{vc}} \varepsilon_a\, E_{\varphi_a}[A_t^a] - |TN|.$$

It can easily be seen from the derivatives of α_a, for all a and β_b, for all b, that the derivative of Λ, as defined by (4.2.4), evaluated at θ is equal to $\tilde{d}_1$, i.e. $\Lambda'(\theta) = \tilde{d}_1$. Under the conditions of Theorem 4.2.5, from Lemma 4.2.2, Lemma 4.2.3 and Lemma 4.2.2, we know that $\Lambda(0) = \Lambda(\theta) = 0$, $\Lambda'(0) < 0$, $\lim_{\gamma \to +\infty} \Lambda(\gamma) = +\infty$, and that Λ is convex. It follows that $\Lambda'(\theta) > 0$. Therefore, $\tilde{d}_1 > 0$.

4.3 The Edge Δ

In this section, we define the set $\Delta \subset S$ large enough such that the probabilities of the transitions from Δ into the interior S° are equal for W and the *free* process W^∞. In other words,

$$\mathbf{1}_{(w \in \Delta)}\, K(w; w')\, \mathbf{1}_{(w' \in S^\circ)} = \mathbf{1}_{(w \in \Delta)}\, K^\infty(w; w')\, \mathbf{1}_{(w' \in S^\circ)}. \tag{4.3.1}$$

We end the section with a discussion of Condition 6 from [McDo99].

As stated in Section 2.5, the set Δ must be defined to include the set of problem points $\mathcal{G}_\Delta$, i.e. $\mathcal{G}_\Delta \subset \Delta$, where

$$\mathcal{G}_\Delta := \{w \in S : \text{ there exists a } b \in TN \text{ such that } q_b = 0 \text{ and } v_b = 0\}. \tag{4.3.2}$$

We will call $\mathcal{G}_\Delta$ the set of gateway points, or the set of lost services. The gateway nomenclature stems from the fact that the transitions (with positive probability) of the *free* process from S into $\blacktriangle \setminus S$ can only occur from the points in $\mathcal{G}_\Delta$.

Away from the set of lost services the kernels K and K^∞ coincide, that is

$$K(w; w') = K^\infty(w; w'), \text{ for all } w \in S \setminus \mathcal{G}_\Delta \text{ and } w \in S. \tag{4.3.3}$$

Also, for all $w \in \mathcal{G}_\Delta$, $K^\infty(w; \blacktriangle \setminus S) = 1$. Thus, if we can define Δ large enough such that the transitions with positive probability, for the process W, from Δ into the interior S° can only occur from $\Delta \setminus \mathcal{G}_\Delta$, then (4.3.1) will hold.

For the ATM-type networks that we are considering, only a finite number of cells can arrive at a node during a time slot. Thus, the following maxima are well-defined,

$$\tilde{\mathfrak{m}}_b := \max\{q'_b - q_b : \text{ for all } w, w' \in S \text{ such that } K(w; w') > 0\}.$$

Define the set Δ as follows,

$$\Delta := \{w \in S : \text{ there exists a } b \in TN \text{ such that } q_b \leq \tilde{\mathfrak{m}}_b\}. \tag{4.3.4}$$

Clearly, $\mathcal{G}_\Delta \subset \Delta$ and $K(w; w') = 0$ for all $w \in \mathcal{G}_\Delta$ and $w' \in S^\circ$. Thus, (4.3.1) holds.

Condition 6 from [McDo99] is $\sum_{w' \in S} \lambda(w') < \infty$, where

$$\lambda(w') := \sum_{w \in \Delta} \pi(w)\, K(w, w')\, h(w')\, \mathbf{1}(w' \in S^\circ).$$

McDonald [McDo99] made the following observation. If the kernels K and K^∞ coincide not just in the interior but also at exits from Δ, that is (4.3.1) holds, then it follows from the definition of the h-transform $\mathcal{K}^\infty$ that the measure λ is equal to

$$\lambda(w') = \sum_{w \in \Delta} \pi(w)\, \mathcal{K}^\infty(w, w')\, h(w)\, \mathbf{1}(w' \in S^\circ).$$

Hence, $\sum_{w' \in \Delta} \lambda(w') < \sum_{w \in \Delta} \pi(w)\, h(w)$, since $\sum_{w' \in S} \mathcal{K}^{\infty}(w, w') \mathbf{1}(w' \in S^{\circ}) \leq 1$. Thus, the following

$$\sum_{w \in S} \pi(w)\, h(w)\, \mathbf{1}(w \in \Delta) < \infty,$$

is a sufficient condition for Condition 6 from [McDo99]. Therefore, Condition C.5 is a sufficient condition for Condition 6 from [McDo99].

Condition C.5 is a statement of the f-regularity of the state space S with respect to the Markov chain W. Chapter 14 from Meyn and Tweedie [MT93] is a good reference on f-regularity. From Theorem 14.3.3 in [MT93], the finiteness of $\pi(f)$ is implied under the following drift criterion, where we take $A = \Delta$.

Condition 4.3.1 *Assume the existence of a non-negative function V, a finite subset $C \subset S$ and positive constants b and c such that*

$$\sum_{w \in S} K(w, w')\, V(w') - V(w) \leq -c\, h(w)\, \mathbf{1}(w \in A) + b\, \mathbf{1}(w \in C),$$

where $A \subseteq S$.

We utilize the latter condition in Section 4.5, to find sufficient conditions for C.5.

4.4 The Twisted Reflected Process

In this section we construct a reflected conjugate process to the twisted free process $\mathcal{W}^{\infty}$. It will be a natural construction. We end the section by linking the V-transience of the set Δ under $\mathcal{K}$ with Condition C.4. See Appendix B for a discussion on the transience of sets.

Let $O_B(w) := \{b \in TN : q_b = 0, v_b = 0\}$ be the set of tagged nodes with lost service slots due to empty queues.

Assume $W[0] = W^{\infty}[0] = w \in S$. If $b \in O_B(w)$, then during the next time slot, i.e. $t = 1$, the *free* total encumbrance ENC^{∞} will be decremented by d_b, but it is not necessarily the case for ENC. We have a lost service slot, if no cells were waiting to

be served at node b the previous time slot. We will denote the lost services as $\zeta_\Delta(w)$, i.e.

$$\zeta_\Delta(w) := \sum_{b \in O_B(w)} d_b.$$

Note, in the interior there are no lost services, i.e. $\zeta_\Delta(w) = 0$ for $w \in S^\circ$.

Choose $w, w' \in S$ such that $K(w; w') > 0$. We must adjust the increment for the total encumbrance to compensate for the lost services. This increment becomes

$$ENC_1 - ENC_0 = \sum_{a \in T_{vc} \cup C_{vc}} \varepsilon_a A_1^a - \sum_{b \in TN} d_b B_1^b + \zeta_\Delta(W[0]).$$

Recall that conditioned on $\hat{W}[0] = \hat{w}$,

$$\sum_{a \in T_{vc} \cup C_{vc}} \varepsilon_a A_1^a - \sum_{b \in TN} d_b B_1^b = \varsigma(e, \hat{w}) - e$$

almost surely for all e, where $\varsigma(e, \hat{w}) - e$ is a constant for fixed $\hat{w}$. Thus conditioned on $W[0] = w = (e, \tilde{w}_2, \ldots, \tilde{w}_r, \hat{w})$, the random variable ENC_1 is almost surely a constant equal to $\varsigma(e, \hat{w}) + \zeta_\Delta(w)$.

We have already shown that the kernel $\widehat{K}_\gamma$ defined as

$$\widehat{K}_\gamma[\hat{w}, \hat{w}'] := E_{\hat{w}} \left[e^{\gamma \left(\sum_{a \in T_{vc} \cup C_{vc}} \varepsilon_a A_1^a - \sum_{b \in TN} d_b B_1^a \right)} \mathbf{1}(\hat{W}[1] = \hat{w}') \right],$$

has a positive right eigenvector $\hat{a}_\gamma(\hat{w})$ with eigenvalue $\Lambda(\gamma)$. This follows from the construction of h_γ in Section 4.2.1. We now want to show that for all $w \in S$,

$$\sum_{w'} K(w, w')\, h_\gamma(w') = e^{\gamma \zeta_\Delta(w)} \exp\{\Lambda(\gamma)\}\, h_\gamma(w).$$

Since $h_\gamma(w) = e^{\gamma e}\, \hat{a}_\gamma(\hat{w})$, it is equivalent to show that

$$\sum_{w'} K(w, w')\, e^{\gamma (e' - e)}\, \hat{a}_\gamma(\hat{w}') = e^{\gamma \zeta_\Delta(w)} \exp\{\Lambda(\gamma)\}\, \hat{a}_\gamma(\hat{w}).$$

We proceed with the proof,

$$\begin{aligned}
&\sum_{w'} K(w,w')\, e^{\gamma(e'-e)}\, \hat{a}_\gamma(\hat{w}') \\
&= \sum_{(e',\hat{w}')} e^{\gamma(e'-e)}\, \hat{a}_\gamma(\hat{w}') \sum_{(\tilde{w}_2,\tilde{w}_3,\ldots,\tilde{w}_r)} K(w,w') \\
&= \sum_{(e',\hat{w}')} E_w \Big[e^{\gamma\left(\sum_{a\in\Psi}\varepsilon_a A_1^a - \sum_{b\in TN} d_b B_1^a + \zeta_\Delta(w)\right)} \times \\
&\qquad\qquad \times \mathbf{1}(ENC_1 = e', \hat{W}[1] = \hat{w}') \Big]\, \hat{a}_\gamma(\hat{w}'),
\end{aligned}$$

where $\Psi = T_{vc} \cup C_{vc}$ is used to compress the notation.

Since conditioned on $W[0] = w = (e, \tilde{w}_2, \ldots, \tilde{w}_r, \hat{w})$, the random variable ENC_1 is almost surely a constant equal to $\varsigma(e,\hat{w}) + \zeta_\Delta(w)$, then we obtain

$$\begin{aligned}
&\sum_{(e',\hat{w}')} E_w \Big[e^{\gamma\left(\sum_{a\in\Psi}\varepsilon_a A_1^a - \sum_{b\in TN} d_b B_1^a + \zeta_\Delta(w)\right)} \times \\
&\qquad\qquad \times \mathbf{1}(e' = \varsigma(e,\hat{w}) + \zeta_\Delta(w))\, \mathbf{1}(\hat{W}[1] = \hat{w}') \Big]\, \hat{a}_\gamma(\hat{w}') \\
&= \sum_{\hat{w}'} E_w \Big[e^{\gamma\left(\sum_{a\in\Psi}\varepsilon_a A_1^a - \sum_{b\in TN} d_b B_1^a + \zeta_\Delta(w)\right)} \mathbf{1}(\hat{W}[1] = \hat{w}') \Big]\, \hat{a}_\gamma(\hat{w}') \times \\
&\qquad \sum_{e'} \mathbf{1}(e' = \varsigma(e,\hat{w}) + \zeta_\Delta(w)) \\
&= e^{\zeta_\Delta(w)} \sum_{\hat{w}'} E_w \Big[e^{\gamma\left(\sum_{a\in\Psi}\varepsilon_a A_1^a - \sum_{b\in TN} d_b B_1^a\right)} \mathbf{1}(\hat{W}[1] = \hat{w}') \Big]\, \hat{a}_\gamma(\hat{w}') \\
&= e^{\gamma\,\zeta_\Delta(w)} \sum_{\hat{w}'} \widehat{K}_\gamma[\hat{w},\hat{w}']\, \hat{a}_\gamma(\hat{w}') \\
&= e^{\gamma\,\zeta_\Delta(w)} \exp\{\Lambda(\gamma)\}\, \hat{a}_\gamma(\hat{w}').
\end{aligned}$$

We summarize the result in the following theorem.

Theorem 4.4.1 *For all $w \in S$,*

$$\sum_{w'} K(w,w')\, h_\gamma(w') = e^{\gamma\,\zeta_\Delta(w)} \exp\{\Lambda(\gamma)\}\, h_\gamma(w).$$

Define $\mathcal{K}$ as the following probability transition kernel on the state space S,

$$\mathcal{K}(w,w') := e^{-\theta\,\zeta_\Delta(w)}\, K(w,w')\, \frac{h(w')}{h(w)}.$$

Note, the kernels $\mathcal{K}$ and $\mathcal{K}^\infty$ coincide in the interior S°. Also, the kernel $\mathcal{K}$ is irreducible. The latter follows from the irreducibility of K. Define $\mathcal{W}$ as the Markov chain generated by the transition kernel $\mathcal{K}$.

Condition 4.4.2 *Suppose there exists a bounded function $V : S \to \mathbb{R}$ such that*

$$\sum_{w' \in S} \mathcal{K}(w; w')\, V(w') - V(w) \leq 0, \text{ for all } x \in S \setminus \Delta. \tag{4.4.1}$$

Suppose, moreover, that there exists a state $w'' \in S^\circ$ such that

$$V(w'') < V(w), \text{ for all } w \in \Delta. \tag{4.4.2}$$

Theorem 4.4.3 *Assume the existence of a V that satisfies Condition 4.4.2. Therefore, Condition C.4 holds.*

Proof : Define for any $A \subset S$,

$$\mathcal{T}_A := \min\{t > 0 : \mathcal{W}[t] \in A\},$$

and for any $B \subset S^\infty$,

$$\mathcal{T}_B^\infty = \min\{t > 0 : \mathcal{W}^\infty[t] \in B\}.$$

It follows from Theorem B.1.3 that $\mathsf{P}_{w''}(\mathcal{T}_\Delta = \infty) > 0$. From the irreducibility of the Markov chain $\mathcal{W}$, it follows that there exists a state $w_0 \in \Delta$, such that $\mathsf{P}_{w_0}(\mathcal{T}_\Delta = \infty) > 0$. The state w_0 does not belong to $\mathcal{G}_\Delta$, since $\mathcal{K}(w; S^\circ) = 0$, for all $w \in \mathcal{G}_\Delta$.

From (4.3.3), it follows that $\mathsf{P}_{w_0}(\mathcal{T}_\Delta = t) = \mathsf{P}_{w_0}(\mathcal{T}_\Delta^\infty = t)$, for all $t \in \mathbb{N}$. Hence, $\mathsf{P}_{w_0}(\mathcal{T}_\Delta^\infty = \infty) > 0$.

If the process $\mathcal{W}^\infty[0] = w_0 \in S \setminus \mathcal{G}_\Delta$, then $\mathcal{W}^\infty$ must first hit the set of gateway points $\mathcal{G}_\Delta \subset \Delta$ before entering $\blacktriangle \setminus \Delta$. Recall, the set $\mathcal{G}_\Delta$ is defined by 4.3.2. It follows that, $\mathcal{T}_\Delta^\infty = \mathcal{T}_\blacktriangle^\infty$, P_{w_0}-almost surely. Therefore, C.4 holds.

□

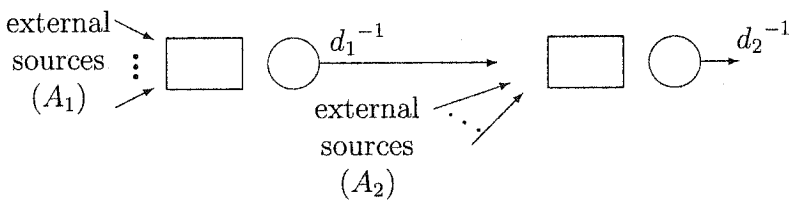

Figure 3: Feedforward Tandem Network : Two Nodes

4.5 Intree Tandem Network (Two Nodes)

In this section, we investigate C.4 and C.5 for a special case of the ATM-type network of Section 2.3. We consider a feedforward intree tandem network. We consider the most simple case, i.e. two nodes, see Figure 3.

The sources can offer cells to either node 1 or node 2. Note, all cells are eventually buffered in a queue at node 2. We will partition the sources into two classes, $A_1 \cup A_2 = A$. The sources in A_b offer cells that arrive at the node b, for $b = 1, 2$.

We consider the case where both nodes are tagged, i.e. $TN = \{1, 2\}$. When source a offers a cell to the network the total encumbrance is increased by

$$\varepsilon_a = \begin{cases} d_1 + d_2, & \text{if } a \in A_1 \\ d_2, & \text{if } a \in A_2 \end{cases}$$

A cell served at node b will decrease the total encumbrance by d_b. Hence, the total encumbrance at the end of time slot t is

$$ENC_t = (d_1 + d_2)\, Q_t^1 + d_2\, Q_t^2,$$

where Q_t^b represents the number of cells buffered at node b at the end of time slot b.

Recall that the edge $\Delta \subset S$ of the boundary $\blacktriangle$ is defined as follows

$$\Delta := \{w \in S : \text{ there exists a } b \in TN \text{ such that } q_b \leq \tilde{\mathfrak{m}}_b\},$$

where

$$\tilde{\mathfrak{m}}_b := \max\{q'_b - q_b : \text{ for all } w, w' \in S \text{ such that } K(w; w') > 0\}.$$

We will consider the following exhaustive collection of subsets of Δ, denoted ${\mathcal{G}_\Delta}^c$, and $\mathcal{G}_\Delta^b$, for $b = 1, 2$. The set of gateway points (also called the set of lost service slots) $\mathcal{G}_\Delta$ is defined by (4.3.2). The subset ${\mathcal{G}_\Delta}^c \subset \Delta$ is the complementary set of $\mathcal{G}_\Delta$ within Δ, i.e. ${\mathcal{G}_\Delta}^c = \Delta \setminus \mathcal{G}$. Note, $K(w, w') = K^\infty(w, w')$, for all $w \in {\mathcal{G}_\Delta}^c$ and $w' \in S$. For $b = 1, 2$, the notation $\mathcal{G}_\Delta^b$ represents the following subsets of $\mathcal{G}_\Delta$

$$\mathcal{G}_\Delta^b := \{w \in \Delta : q_b = 0, v_b = 0\}.$$

We say that the three sets form an exhaustive collection of subsets of Δ since

$$\Delta = {\mathcal{G}_\Delta}^c \cup \mathcal{G}_\Delta^1 \cup \mathcal{G}_\Delta^2.$$

Our goal is to construct a Lyapounov function $V : S \to \mathbb{R}_+$ that satisfies Condition 4.3.1. It will be of the form $V(w) = V_1(w) + V_2(w)$, where

$$V_b(w) = \exp\{\theta\, e - \nu_b\, q_b\}\, \hat{a}^{*b}_{\nu_b}(\hat{w}),$$

$\nu_b > 0$ and $\hat{a}^{*b}_{\nu_b} > 0$ for $b = 1, 2$. Note, $\hat{a}^{*b}_s$ are positive functions on $\hat{S}$ with parameter s. The construction of these functions is given in the next subsection.

We will need the following extrema. Define $\underline{\xi}(s), \overline{\xi}(s) \in \mathbb{R}_+$, as

$$\underline{\xi}(s) := \min\left\{\frac{\hat{a}^{*b}_s(\hat{w})}{\hat{a}_\theta(\hat{w})} : \text{ for all } \hat{w} \in \hat{S}, \text{ for all } b \in \{1, 2\}\right\}, \tag{4.5.1}$$

and

$$\overline{\xi}(s) := \max\left\{\frac{\hat{a}^{*b}_s(\hat{w})}{\hat{a}_\theta(\hat{w})} : \text{ for all } \hat{w} \in \hat{S}, \text{ for all } b \in \{1, 2\}\right\}. \tag{4.5.2}$$

Using (4.5.1) and (4.5.2), we obtain the following inequalities, for all $w \in S^\infty$,

$$\exp\{\theta\, e\}\, \hat{a}^{*b}_s(\hat{w}) \geq \underline{\xi}(s)\, \exp\{\theta\, e\}\, \hat{a}_\theta(\hat{w}) = \underline{\xi}(s)\, h(w), \tag{4.5.3}$$

and

$$\exp\{\theta\, e\}\, \hat{a}^{*b}_s(\hat{w}) \leq \overline{\xi}(s)\, \exp\{\theta\, e\}\, \hat{a}_\theta(\hat{w}) = \overline{\xi}(s)\, h(w). \tag{4.5.4}$$

For all $w \in \mathcal{G}_\Delta^b$, $q_b = 0$, thus $V_b(w) = \exp\{\theta\, e\}\, \hat{a}^{*b}_{\nu_b}(\hat{w})$ is bounded above and below by multiples of $h(w)$. In other words, on the set of problem points $\mathcal{G}_\Delta$, the Lyapounov function V is "similar" to h.

In the next subsection, we will construct V_1^s and V_2^s such that they will be eigenvectors for K^∞ with respective eigenvalues $\exp(\Lambda_1(s))$ and $\exp(\Lambda_2(s))$. We start by stating a few lemmas and the main result. The proof of the lemmas are also found in the following subsection.

Lemma 4.5.1 *The functions V_b^s, $b = 1, 2$, defined as follows*

$$V_b^s(w) := \exp\{\theta\, e - s\, q_b\}\, \hat{a}_s^{*b}(\hat{w}), \tag{4.5.5}$$

where $\hat{a}_s^{*1}$, and $\hat{a}_s^{*2}$ are defined by (4.5.23) and (4.5.30), respectively, are positive eigenvectors for K^∞. We denote their respective eigenvalues as $\exp(\Lambda_b(s))$.

Lemma 4.5.2 *The functions V_b^s, from (4.5.5), $b = 1, 2$, satisfy the following equations :*

$$K\, V_b^s(w) = \exp(\Lambda_b(s))\, V_b^s(w), \ \text{if } w \in S \setminus \mathcal{G}_\Delta \tag{4.5.6}$$

$$K\, V_1^s(w) = \begin{cases} e^{\theta\, d_2}\, \exp(\Lambda_1(s))\, V_1^s(w), & \text{if } w \in \mathcal{G}_\Delta^2 \text{ and } q_1 > 0, \\ e^{\theta\, d_1 - s}\, \exp(\Lambda_1(s))\, V_1^s(w), & \text{if } w \in \mathcal{G}_\Delta^1 \text{ and } q_2 > 0, \end{cases} \tag{4.5.7}$$

and

$$K\, V_2^s(w) = \begin{cases} e^{\theta\, d_1 + s}\, \exp(\Lambda_2(s))\, V_2^s(w), & \text{if } w \in \mathcal{G}_\Delta^1 \text{ and } q_2 > 0, \\ e^{\theta\, d_2 - s}\, \exp(\Lambda_2(s))\, V_2^s(w), & \text{if } w \in \mathcal{G}_\Delta^2 \text{ and } q_1 > 0. \end{cases} \tag{4.5.8}$$

Lemma 4.5.3 *Assume that $V_b^s(w) = \exp\{\theta\, e - s\, q_b\}\, \hat{a}_s^{*b}(\hat{w})$, as defined by (4.5.5), is a positive eigenvector for K^∞ with eigenvalue $\exp(\Lambda_b(s))$, then*

$$\mathcal{V}_b^s(w) := \exp\{-s\, q_b\}\, \frac{\hat{a}_s^{*b}(\hat{w})}{\hat{a}_\theta(\hat{w})} \tag{4.5.9}$$

is an eigenvector for $\mathcal{K}^\infty$ with eigenvalue $\exp(\Lambda_b(s))$.

Lemma 4.5.4 *Assume*

$$\sum_{a \in A_1} E_{\varphi_a}[A_t^a] > \frac{1}{d_1} \quad \text{and} \quad \sum_{a \in A_2} E_{\varphi_a}[A_t^a] + \frac{1}{d_1} > \frac{1}{d_2}.$$

For $b = 1, 2$, there exists $s_b > 0$ such that $\Lambda_b(s_b) < 0$.

Theorem 4.5.5 *Under the conditions of Lemma 4.5.4, we can choose $\nu_b > 0$ such that $\Lambda_b(\nu_b) < 0$.*

a) The function

$$\mathcal{V} := \mathcal{V}_1^{\nu_1} + \mathcal{V}_2^{\nu_2},$$

where $\mathcal{V}_b^s$ is defined by (4.5.9), can be used as the bounded Lyapounov function V from Condition 4.4.2. Therefore, C.4 holds.

b) If $\nu_b > 0$, b=1,2, can be chosen such that $\min_{b\in\{1,2\}} \left\{ \frac{\nu_b}{d_b} \right\} > \theta$, then the function

$$V := V_1^{\nu_1} + V_2^{\nu_2}$$

can be used as the non-negative Lyapounov function V from Condition 4.3.1, where the strictly positive constants ϵ and b and the finite subset $C \subset S$ are chosen appropriately. Therefore, C.5 holds.

Proof : a) Since $\exp(\Lambda_b(\nu_b)) < 1$, for $b = 1, 2$, then, for all $w \in S^\infty$,

$$\begin{aligned} \mathcal{K}^\infty \mathcal{V}(w) &= \mathcal{K}^\infty \mathcal{V}_1^{\nu_1}(w) + \mathcal{K}^\infty \mathcal{V}_2^{\nu_2}(w) \\ &= e^{\Lambda_1(\nu_1)} \mathcal{V}_1^{\nu_1}(w) + e^{\Lambda_2(\nu_2)} \mathcal{V}_2^{\nu_2}(w) \\ &\leq \mathcal{V}_1^{\nu_1}(w) + \mathcal{V}_2^{\nu_2}(w) = \mathcal{V}(w). \end{aligned}$$

Since $\mathcal{K}(w; w') = \mathcal{K}^\infty(w, w')$, for all $w \in S^\circ$, and $w' \in S$, then

$$\mathcal{K}\mathcal{V}(w) \leq \mathcal{V}(w), \text{ for all } w \in S \setminus \Delta.$$

It remains to show that there exists a state $w'' \in S^\circ$, such that $\mathcal{V}(w'') < \mathcal{V}(w)$, for all $w \in \Delta$. Since, on Δ there exists a $b_0 \in TN = \{1, 2\}$ such that $q_{b_0} \leq \tilde{\mathfrak{m}}_{b_0}$, see (4.3.4). Using $\underline{\xi}(\nu_{b_0})$ from (4.5.1), we have for all $w \in \Delta$

$$\begin{aligned} \mathcal{V}(w) &= e^{-\nu_1 q_1} \frac{\hat{a}_{\nu_1}^{*1}(\hat{w})}{\hat{a}_\theta(\hat{w})} + e^{-\nu_2 q_2} \frac{\hat{a}_{\nu_2}^{*2}(\hat{w})}{\hat{a}_\theta(\hat{w})} \\ &\geq e^{-\nu_{b_0} q_{b_0}} \frac{\hat{a}_{\nu_{b_0}}^{*b_0}(\hat{w})}{\hat{a}_\theta(\hat{w})} \\ &\geq \underline{\xi}(\nu_{b_0}) \exp(-\nu_{b_0} \tilde{\mathfrak{m}}_{b_0}) \\ &\geq \min_{b\in\{1,2\}} \left\{ \underline{\xi}(\nu_b) \exp(-\nu_b \tilde{\mathfrak{m}}_b) \right\} > 0. \end{aligned}$$

It follows that such a $w'' \in S^\circ$ exists, since $\mathcal{V}(w) \to 0$ as $q_1 \to \infty$ and $q_2 \to \infty$.

b) We will consider the following three cases: *i*) $w \in {\mathcal{G}_\Delta}^c$; *ii*) $w \in \mathcal{G}_\Delta^1$; and *iii*) $w \in \mathcal{G}_\Delta^2$.

i) Assume that $w \in {\mathcal{G}_\Delta}^c$. We will show that there exists a positive constant ϵ_1 such that

$$KV(w) - V(w) \leq -\epsilon_1 \, h(w).$$

Using (4.5.6), we obtain

$$\begin{aligned} KV(w) - V(w) &= K\left(V_1^{\nu_1}(w) + V_2^{\nu_2}(w)\right) - \left(V_1^{\nu_1}(w) + V_2^{\nu_2}(w)\right) \\ &= \sum_{b=1}^{2} \left\{KV_b^{\nu_b}(w) - V_b^{\nu_b}(w)\right\} \\ &= \sum_{b=1}^{2} \left\{e^{\Lambda_b(\nu_b)} V_b^{\nu_b}(w) - V_b^{\nu_b}(w)\right\}. \end{aligned}$$

From the definition of $V_b^{\nu_b}$, see (4.5.5), it follows that

$$KV(w) - V(w) = e^{\theta\, e} \sum_{b=1}^{2} \left\{e^{-\nu_b\, q_b} \hat{a}_{\nu_b}^{*b}(\hat{w}) \left(e^{\Lambda_b(\nu_b)} - 1\right)\right\} \tag{4.5.10}$$

Since $\exp(\Lambda_b(\nu_b)) < 1$, for $b = 1, 2$, then both terms in the sum on the right-hand-side of (4.5.10) are strictly negative. It follows that

$$KV(w) - V(w) \leq e^{\theta\, e}\, e^{-\nu_{b_0}\, q_{b_0}} \hat{a}_{\nu_b}^{*b_0}(\hat{w}) \left(e^{\Lambda_{b_0}(\nu_{b_0})} - 1\right), \tag{4.5.11}$$

where $b_0 \in TN = \{1, 2\}$ is chosen such that $q_{b_0} \leq \tilde{\mathfrak{m}}_{b_0}$. Such a b_0 exists since $w \in {\mathcal{G}_\Delta}^c \subset \Delta$, see (4.3.4). Thus,

$$e^{-\nu_{b_0}\, q_{b_0}} \geq e^{-\nu_{b_0}\, \tilde{\mathfrak{m}}_{b_0}} \geq \min_{b \in \{1,2\}} \left\{e^{-\nu_b\, \tilde{\mathfrak{m}}_b}\right\} > 0.$$

From the latter and the fact that the right-hand-side of (4.5.11) is negative, we obtain

$$KV(w) - V(w) \leq e^{\theta\, e}\, \hat{a}_{\nu_b}^{*b_0}(\hat{w}) \left(e^{\Lambda_{b_0}(\nu_{b_0})} - 1\right) \times \min_{b \in \{1,2\}} \left\{e^{-\nu_b\, \tilde{\mathfrak{m}}_b}\right\}. \tag{4.5.12}$$

Using (4.5.3), we get the following inequalities

$$e^{\theta\, e}\, \hat{a}_{\nu_b}^{*b_0}(\hat{w}) \geq \underline{\xi}(\nu_{b_0})\, h(w) \geq h(w) \min_{b \in \{1,2\}} \left\{\underline{\xi}(\nu_b)\right\} > 0.$$

From the latter and the fact that the right-hand-side of (4.5.12) is negative, we obtain

$$KV(w) - V(w) \leq h(w)\,(e^{\Lambda_{b_0}(\nu_{b_0})} - 1) \times \min_{b\in\{1,2\}} \left\{e^{-\nu_b\,\tilde{m}_b}\right\} \times \min_{b\in\{1,2\}} \left\{\underline{\xi}(\nu_b)\right\}.$$

Let ϵ_1 be the following positive constant,

$$\epsilon_1 = \min_{b\in\{1,2\}} \left\{|e^{\Lambda_b(\nu_b)} - 1|\right\} \times \min_{b\in\{1,2\}} \left\{e^{-\nu_b\,\tilde{m}_b}\right\} \times \min_{b\in\{1,2\}} \left\{\underline{\xi}(\nu_b)\right\} > 0.$$

Therefore,

$$KV(w) - V(w) \leq -\epsilon_1\, h(w).$$

We have proved case (i). We now proceed to the other two cases.

For the cases (ii) and (iii), we will need $C \subset S$ defined as

$$C := \{w \in S : q_1 \leq c_1 \text{ and } q_2 \leq c_2\}. \tag{4.5.13}$$

The strictly positive constants c_1 and c_2 are going to be defined as we proceed with the proof. Note, the subset C is finite.

ii) Assume that $w \in \mathcal{G}_\Delta^1$. We will prove that for $w \notin C$, there exists a strictly positive constant ϵ_2 such that

$$KV(w) - V(w) \leq -\epsilon_2\, h(w).$$

Since $w \notin C$, then for all $w \in \mathcal{G}_\Delta^1$, $q_1 = 0$, thus $q_2 > c_2 > 0$.

Using (4.5.7) and (4.5.8), we obtain

$$\begin{aligned} & KV(w) - V(w) \\ = \;& KV_1^{\nu_1}(w) - V_1^{\nu_1}(w) + KV_2^{\nu_2}(w) - V_2^{\nu_2}(w) \\ = \;& \left(e^{\theta\, d_1 - \nu_1}\, e^{\Lambda_1(\nu_1)} - 1\right) V_1^{\nu_1}(w) + \left(e^{\theta\, d_1 + \nu_2}\, e^{\Lambda_2(\nu_2)} - 1\right) V_2^{\nu_2}(w). \end{aligned} \tag{4.5.14}$$

We will consider the two terms from (4.5.14) separately. The idea is as follows. We will show that the first term is smaller than some negative multiple of h, and that the second term (which can possibly be positive) will be insignificant if we choose c_2 large enough.

We start with the first term from (4.5.14),

$$\left(e^{\theta\, d_1-\nu_1}\, e^{\Lambda_1(\nu_1)}-1\right)\, V_1^{\nu_1}(w)=\left(e^{\theta\, d_1-\nu_1}\, e^{\Lambda_1(\nu_1)}-1\right)\, e^{\theta\, e}\, \hat{a}_{\nu_1}^{*1}(\hat{w}), \tag{4.5.15}$$

since $q_1 = 0$. From the statement of the theorem, $\theta\, d_1 - \nu_1 < 0$ and $\Lambda_1(\nu_1) < 0$. Hence, $e^{\theta\, d_1-\nu_1}\, e^{\Lambda_1(\nu_1)} - 1 < 0$. Using (4.5.3) and the fact that (4.5.15) is negative, we obtain

$$\left(e^{\theta\, d_1-\nu_1}\, e^{\Lambda_1(\nu_1)}-1\right)\, V_1^{\nu_1}(w) \leq \left(e^{\theta\, d_1-\nu_1}\, e^{\Lambda_1(\nu_1)}-1\right)\, \underline{\xi}(\nu_1)\, h(w).$$

Let ϵ_{21} be the following positive constant,

$$\epsilon_{21} := -\underline{\xi}(\nu_1)\, \left(e^{\theta\, d_1-\nu_1}\, e^{\Lambda_1(\nu_1)}-1\right) > 0.$$

Therefore, we obtain the following bound for the first term from (4.5.14),

$$\left(e^{\theta\, d_1-\nu_1}\, e^{\Lambda_1(\nu_1)}-1\right)\, V_1^{\nu_1}(w) \leq -\epsilon_{21}\, h(w). \tag{4.5.16}$$

Now we consider the second term from (4.5.14),

$$\begin{aligned}
& \left(e^{\theta\, d_1+\nu_2}\, e^{\Lambda_2(\nu_2)}-1\right)\, V_2^{\nu_2}(w) \\
= \;& \left(e^{\theta\, d_1+\nu_2}\, e^{\Lambda_2(\nu_2)}-1\right)\, e^{\theta\, e-\nu_2\, q_2}\, \hat{a}_{\nu_2}^{*2}(\hat{w}) \\
\leq \;& \left|e^{\theta\, d_1+\nu_2}\, e^{\Lambda_2(\nu_2)}-1\right|\, e^{-\nu_2\, q_2}\, e^{\theta\, e}\, \hat{a}_{\nu_2}^{*2}(\hat{w}) \\
\leq \;& \left|e^{\theta\, d_1+\nu_2}\, e^{\Lambda_2(\nu_2)}-1\right|\, e^{-\nu_2\, q_2}\, \overline{\xi}(\nu_2)\, h(w), \qquad \text{using (4.5.4)} \\
\leq \;& \left|e^{\theta\, d_1+\nu_2}\, e^{\Lambda_2(\nu_2)}-1\right|\, e^{-\nu_2\, c_2}\, \overline{\xi}(\nu_2)\, h(w), \qquad \text{since } q_2 > c_2 \\
= \;& \epsilon_{22}\, h(w),
\end{aligned}$$

where we define $\epsilon_{22} > 0$ as follows

$$\epsilon_{22} := \left|e^{\theta\, d_1+\nu_2}\, e^{\Lambda_2(\nu_2)}-1\right|\, e^{-\nu_2\, c_2}\, \overline{\xi}(\nu_2).$$

Therefore, we obtain the following bound for the second term from (4.5.14),

$$\left(e^{\theta\, d_1}\, e^{\Lambda_2(\nu_2)}-1\right)\, V_2^{\nu_2}(w) \leq \epsilon_{22}\, h(w). \tag{4.5.17}$$

From (4.5.14), (4.5.16) and (4.5.17), we obtain

$$KV(w) - V(w) \leq (-\epsilon_{21} + \epsilon_{22})\, h(w).$$

To define ϵ_{22} properly, we must fixed the value of the constant c_2. We will choose c_2 large enough such that $\epsilon_{22} < \epsilon_{21}$. This is possible since as $c_2 \to \infty$, ϵ_{22} as a function of c_2 tends to zero.

Define $\epsilon_2 := \epsilon_{21} - \epsilon_{22} > 0$. Therefore, for all $w \in \mathcal{G}_\Delta^1 \cap (S \setminus C)$,

$$KV(w) - V(w) \leq -\epsilon_2\, h(w).$$

This completes the proof of case (ii).

iii) Assume that $w \in \mathcal{G}_\Delta^2$. We will prove that for $w \notin C$, there exists a strictly positive constant ϵ_3 such that

$$KV(w) - V(w) \leq -\epsilon_3\, h(w).$$

Since $w \notin C$, then for all $w \in \mathcal{G}_\Delta^2$, $q_2 = 0$, thus $q_1 > c_1 > 0$.

Using (4.5.7) and (4.5.8), we obtain

$$\begin{aligned} & KV(w) - V(w) \\ = \;& KV_1^{\nu_1}(w) - V_1^{\nu_1}(w) + KV_2^{\nu_2}(w) - V_2^{\nu_2}(w) \\ = \;& \left(e^{\theta\, d_2}\, e^{\Lambda_1(\nu_1)} - 1\right)\, V_1^{\nu_1}(w) + \left(e^{\theta\, d_2 - \nu_2}\, e^{\Lambda_2(\nu_2)} - 1\right)\, V_2^{\nu_2}(w). \end{aligned} \tag{4.5.18}$$

We will consider the two terms from (4.5.18) separately. We will proceed as in case (ii). We will show that the second term is smaller than some negative multiple of h, and that the first term (which can possibly be positive) will be insignificant if we choose c_1 large enough.

We start with the second term from (4.5.18),

$$\left(e^{\theta\, d_2 - \nu_2}\, e^{\Lambda_2(\nu_2)} - 1\right)\, V_2^{\nu_2}(w) = \left(e^{\theta\, d_2 - \nu_2}\, e^{\Lambda_2(\nu_2)} - 1\right)\, e^{\theta\, e}\, \hat{a}_{\nu_2}^{*2}(\hat{w}), \tag{4.5.19}$$

since $q_2 = 0$. From the statement of the theorem, $\theta\, d_2 - \nu_2 < 0$ and $\Lambda_2(\nu_2) < 0$. Hence, $e^{\theta\, d_2 - \nu_2}\, e^{\Lambda_2(\nu_2)} - 1 < 0$. Using (4.5.3) and the fact that (4.5.19) is negative, we obtain

$$\left(e^{\theta\, d_2 - \nu_2}\, e^{\Lambda_2(\nu_2)} - 1\right)\, V_2^{\nu_2}(w) \leq \left(e^{\theta\, d_2 - \nu_2}\, e^{\Lambda_2(\nu_2)} - 1\right)\, \underline{\xi}(\nu_2)\, h(w).$$

Let ϵ_{31} be the following positive constant,

$$\epsilon_{31} := -\underline{\xi}(\nu_2)\, \left(e^{\theta\, d_2 - \nu_2}\, e^{\Lambda_2(\nu_2)} - 1\right) > 0.$$

Therefore, we obtain the following bound for the second term from (4.5.18),

$$\left(e^{\theta d_2 - \nu_2}\, e^{\Lambda_2(\nu_2)} - 1\right)\, V_2^{\nu_2}(w) \leq -\epsilon_{31}\, h(w). \tag{4.5.20}$$

Now we consider the first term from (4.5.18),

$$\begin{aligned}
& \left(e^{\theta d_2}\, e^{\Lambda_1(\nu_1)} - 1\right)\, V_1^{\nu_1}(w) \\
= \;& \left(e^{\theta d_2}\, e^{\Lambda_1(\nu_1)} - 1\right)\, e^{\theta e - \nu_1 q_1}\, \hat{a}_{\nu_1}^{*1}(\hat{w}) \\
\leq \;& \left|e^{\theta d_2}\, e^{\Lambda_1(\nu_1)} - 1\right|\, e^{-\nu_1 q_1}\, e^{\theta e}\, \hat{a}_{\nu_1}^{*1}(\hat{w}) \\
\leq \;& \left|e^{\theta d_2}\, e^{\Lambda_1(\nu_1)} - 1\right|\, e^{-\nu_1 q_1}\, \overline{\xi}(\nu_1)\, h(w), \qquad \text{using (4.5.4)} \\
\leq \;& \left|e^{\theta d_2}\, e^{\Lambda_1(\nu_1)} - 1\right|\, e^{-\nu_1 c_1}\, \overline{\xi}(\nu_1)\, h(w), \qquad \text{since } q_1 > c_1 \\
= \;& \epsilon_{32}\, h(w),
\end{aligned}$$

where we define $\epsilon_{32} > 0$ as follows

$$\epsilon_{32} := \left|e^{\theta d_2}\, e^{\Lambda_1(\nu_1)} - 1\right|\, e^{-\nu_1 c_1}\, \overline{\xi}(\nu_1).$$

Therefore, we obtain the following bound for the first term from (4.5.18),

$$\left(e^{\theta d_1}\, e^{\Lambda_2(\nu_2)} - 1\right)\, V_2^{\nu_2}(w) \leq \epsilon_{32}\, h(w). \tag{4.5.21}$$

From (4.5.18), (4.5.20) and (4.5.21), we obtain

$$KV(w) - V(w) \leq (-\epsilon_{31} + \epsilon_{32})\, h(w).$$

To define ϵ_{32} properly, we must fixed the value of the constant c_1. We will choose c_1 large enough such that $\epsilon_{32} < \epsilon_{31}$. This is possible since as $c_1 \to \infty$, ϵ_{32} as a function of c_1 tends to zero.

Define $\epsilon_3 := \epsilon_{31} - \epsilon_{32} > 0$. Therefore, for all $w \in \mathcal{G}_\Delta^2 \cap (S \setminus C)$,

$$KV(w) - V(w) \leq -\epsilon_3\, h(w).$$

This completes the proof of case *(iii)*.

Let $\epsilon = \min_{i \in \{1,2,3\}} \{\epsilon_i\} > 0$. We have shown that for all $w \in \Delta$,

$$KV(w) - V(w) \leq -\epsilon\, h(w) + b\, \mathbf{1}(w \in C),$$

where b is a strictly positive constant. Such a b exists. This follows from the fact that C is a finite subset, which implies that $KV(w) - V(w)$ is bounded on C.

□

4.5.1 Construction of the Lyapounov Functions

In this subsection, we prove Lemma 4.5.1 and Lemma 4.5.4. The proof for Lemma 4.5.2 and Lemma 4.5.3 are found in Appendix A. We will concentrate on one node at a time.

Let us start with the construction of V_1^s. Define

$$V_1^s(w) := \exp\{\theta\, e - s\, q_1\}\, \hat{a}_s^{*1}(\hat{w}), \tag{4.5.22}$$

where

$$\hat{a}_s^{*1}(\hat{w}) := \prod_{a\in A_1} \hat{a}_s^{*a}(m_a, u_a) \prod_{a\in A_2} \hat{a}_\theta^a(m_a, u_a) \times \hat{a}_s^{*1}(v_1) \times \hat{a}_{-\theta}^2(v_2). \tag{4.5.23}$$

Note, the notation $\hat{a}_s^{*\cdot}$ is being used to denote different functions. Although, its input should indicate which of the functions it is representing. The functions $\hat{a}_s^{*a}(m_a, u_a)$ are defined in Lemma 3.5.1 and the functions $\hat{a}_s^{*b}(v_b)$ are defined in Lemma 3.6.2.

In the next few pages we will prove the following result.

Lemma 4.5.6 ***Part i:*** *The function V_1^s as defined by (4.5.22) is a positive eigenvector for K^∞ with eigenvalue* $\exp(\Lambda_1(s))$ *as defined by (4.5.26).* ***Part ii:*** *If*

$$\sum_{a\in A_1} E_{\varphi_a}[A_t^a] > \frac{1}{d_1},$$

then there exists a real number $s_1 > 0$ such that $\Lambda_1(s_1) < 0$.

The process $Q_t^{b\,\infty}$ will represent the free process associated to the queueing process Q_t^b by ignoring the reflections in Δ. The increments of ENC_t^∞ and $Q_t^{1\,\infty}$ are

$$\Delta\, ENC_t^\infty = \sum_{a\in A_1\cup A_2} \varepsilon_a\, A_t^a - \sum_{b\in TN} d_b\, B_t^b, \tag{4.5.24}$$

and

$$\Delta\, Q_t^{1\,\infty} = \left(\sum_{a\in A_1} A_t^a\right) - B_t^1. \tag{4.5.25}$$

Recall, A_b is the index set for the sources that directly offer cells to node b, A_t^a is the indicator function which is 1 when source a offers a cell during the time slot t, and

B_t^b is the indicator function which is 1 when a cell is served at node b during the time slot t.

Define the operator

$$\widehat{K}_s^{1*}[\hat{w}, \hat{w}'] := E_{\hat{w}} \left[\exp\left\{ \theta \, \Delta \, ENC_1^{\infty} - s \, \Delta \, Q_1^{1\infty} \right\} \, \mathbf{1}(\hat{W}[1] = \hat{w}') \right].$$

From (4.5.24) and (4.5.25), we obtain

$$\begin{aligned} & \theta \Delta \, ENC_1^{\infty} - s \, \Delta \, Q_1^{1\infty} \\ = \; & \theta \left(\sum_{a \in A_1 \cup A_2} \varepsilon_a \, A_1^a - \sum_{b \in TN} d_b \, B_1^b \right) - s \left(\left(\sum_{a \in A_1} A_1^a \right) - B_1^1 \right) \\ = \; & \sum_{a \in A_1} (\theta \, \varepsilon_a - s) \, A_1^a + \sum_{a \in A_2} \theta \, \varepsilon_a \, A_1^a + (-\theta \, d_1 + s) \, B_1^1 - \theta \, d_2 \, B_1^2. \end{aligned}$$

Which implies, by the independence of the sources and the nodes, that

$$\begin{aligned} & \widehat{K}_s^{1*}[\hat{w}, \hat{w}'] \\ = \; & \prod_{a \in A_1} G_s^{a*}(m_a, u_a; m'_a, u'_a) \times \prod_{a \in A_2} G_\theta^a(m_a, u_a; m'_a, u'_a) \times H_s^{1*}(v_1, v'_1) \times H_\theta^2(v_2, v'_2), \end{aligned}$$

where G_θ^a is G_γ^a at $\gamma = \theta$ as defined in Lemma 3.4.3, G_s^{a*} is defined in Lemma 3.5.1, H_θ^2 is H_γ^b at $b = 2$ and $\gamma = \theta$ as defined in Lemma 3.6.1, and H_s^{1*} is defined in Lemma 3.6.2.

It follows from Lemma 3.5.1, Lemma 3.4.3, Lemma 3.6.2 and Lemma 3.6.1 that the kernel $\widehat{K}_s^{1*}$ has a positive right eigenvector $\hat{a}_s^{*1}$ as follows

$$\hat{a}_s^{*1}(\hat{w}) = \prod_{a \in A_1} \hat{a}_s^{*a}(m_a, u_a) \prod_{a \in A_2} \hat{a}_\theta^a(m_a, u_a) \times \hat{a}_s^{*1}(v_1) \times \hat{a}_{-\theta}^2(v_2),$$

with eigenvalue $\exp\{\Lambda_1(s)\}$, where

$$\Lambda_1(s) := \sum_{a \in A_1} -\alpha_a^*(s) + \sum_{a \in A_2} -\alpha_a(\theta) - \beta_1^*(s) - \beta_2(-\theta). \tag{4.5.26}$$

This implies that

$$\sum_{w'} K^{\infty}(w, w') V_1^s(w') = \exp\{\Lambda_1(s)\} \, V_1^s(w).$$

This proves *Part i* of Lemma 4.5.6.

We will now show that

$$\sum_{a\in A_1} E_{\varphi_a}[A_t^a] > \frac{1}{d_1} \tag{4.5.27}$$

is a sufficient condition for the existence of a positive real number s_1 such that $\Lambda_1(s_1) < 0$. Note, this condition means that the mean arrival rate at node 1 must be greater than the mean service rate at node 1. Evaluating Λ_1 at $s = 0$, we obtain

$$\begin{aligned}
\Lambda_1(0) &= \sum_{a\in A_1} -\alpha_a^*(0) + \sum_{a\in A_2} -\alpha_a(\theta) - \beta_1^*(0) - \beta_2(-\theta) \\
&= \sum_{a\in A_1} -\alpha_a(\theta) + \sum_{a\in A_2} -\alpha_a(\theta) - \beta_1(-\theta) - \beta_2(-\theta) \qquad (4.5.28) \\
&= \Lambda(\theta) = 0.
\end{aligned}$$

The equality (4.5.28) follows from the definitions of $\alpha_a^*(s)$ and $\beta_1^*(s)$, see Lemma 3.5.1 and Lemma 3.6.2, respectively.

By Lemma 3.5.1 and Lemma 3.6.2, the derivative of Λ_1 at $s = 0$ is equal to

$$\Lambda_1'(0) = \sum_{a\in A_1} -E_{\varphi_a}[A_t^a] + \frac{1}{d_1}.$$

Under the inequality (4.5.27), this derivative is negative. Therefore, there exists a positive real number s_1 such that $\Lambda_1(s_1) < 0$. This proves *Part ii* of Lemma 4.5.6.

We end this subsection with the construction of V_2^s. It will be similar to the construction for V_1^s.

Define

$$V_2^s(w) := \exp\{\theta\, e - s\, q_2\}\, \hat{a}_s^{*2}(\hat{w}), \tag{4.5.29}$$

where

$$\hat{a}_s^{*2}(\hat{w}) := \prod_{a\in A_2} \hat{a}_s^{*a}(m_a, u_a) \prod_{a\in A_1} \hat{a}_\theta^a(m_a, u_a) \times \hat{a}_s^{*2}(v_2) \times \hat{a}_{-s}^{*1}(v_1). \tag{4.5.30}$$

In the next few pages we will prove the following result.

Lemma 4.5.7 ***Part i:*** *The function V_2^s as defined by (4.5.29) is a positive eigenvector for K^∞ with eigenvalue* $\exp(\Lambda_2(s))$ *as defined by (4.5.32).* ***Part ii:*** *If*

$$\sum_{a\in A_2} E_{\varphi_a}[A_t^a] + \frac{1}{d_1} > \frac{1}{d_2},$$

then there exists a real number $s_2 > 0$ *such that* $\Lambda_2(s_2) < 0$.

The increments for the *free* process $Q_t^{2\infty}$ are

$$\Delta Q_t^{2\infty} = \left(\sum_{a\in A_2} A_t^a\right) - B_t^2 + B_t^1, \tag{4.5.31}$$

where $B_t^1 = 1$, if there is an arrival from node 1. Note, the serviced cells from node 1 become arrivals at node 2.

Define the operator

$$\widehat{K}_s^{2*}[\hat{w}, \hat{w}'] := E_{\hat{w}}\left[\exp\left\{\theta\,\Delta\, ENC_1^{\infty} - s\,\Delta\, Q_1^{2\infty}\right\}\, \mathbf{1}(\hat{W}[1] = \hat{w}')\right].$$

From (4.5.24) and (4.5.31), we obtain

$$\begin{aligned}
&\theta\Delta\, ENC_1^{\infty} - s\,\Delta\, Q_1^{2\infty} \\
&= \theta\left(\sum_{a\in A_1\cup A_2} \varepsilon_a\, A_1^a - \sum_{b\in TN} d_b\, B_1^b\right) - s\left(\left(\sum_{a\in A_2} A_1^a\right) - B_1^2 + B_1^1,\right) \\
&= \sum_{a\in A_2} (\theta\,\varepsilon_a - s)\, A_1^a + \sum_{a\in A_1} \theta\,\varepsilon_a\, A_1^a + (-\theta\, d_2 + s)\, B_1^2 + (-\theta\, d_1 - s)\, B_1^1.
\end{aligned}$$

Which implies, by the independence of the sources and the nodes, that

$$\begin{aligned}
&\widehat{K}_s^{2*}[\hat{w}, \hat{w}'] \\
&= \prod_{a\in A_2} G_s^{a*}(m_a, u_a; m_a', u_a') \times \prod_{a\in A_1} G_\theta^a(m_a, u_a; m_a', u_a') \times H_s^{2*}(v_2, v_2') \times H_{-s}^{1*}(v_1, v_1').
\end{aligned}$$

where G_s^{a*}, G_θ^a, H_s^{2*} and H_{-s}^{1*} are defined in Lemma 3.5.1, Lemma 3.4.3, Lemma 3.6.2 and Lemma 3.6.2, respectively.

It follows from Lemma 3.5.1, Lemma 3.4.3, Lemma 3.6.2 and Lemma 3.6.1 that the kernel $\widehat{K}_s^{2*}$ has a positive right eigenvector $\hat{a}_s^{*2}$ as follows

$$\hat{a}_s^{*2}(\hat{w}) = \prod_{a\in A_1} \hat{a}_s^{*a}(m_a, u_a) \prod_{a\in A_2} \hat{a}_\theta^a(m_a, u_a) \times \hat{a}_{-s}^{*1}(v_1) \times \hat{a}_s^{*2}(v_2),$$

with eigenvalue $\exp\{\Lambda_2(s)\}$, where

$$\Lambda_2(s) := \sum_{a\in A_2} -\alpha_a^*(s) + \sum_{a\in A_1} -\alpha_a(\theta) - \beta_2^*(s) - \beta_1^*(-s). \tag{4.5.32}$$

This implies that

$$\sum_{w'} K^{\infty}(w,w')V_2^s(w') = \exp\{\Lambda_2(s)\}\, V_2^s(w).$$

This proves *Part i* from Lemma 4.5.7.

We now will show that

$$\sum_{a\in A_2} E_{\varphi_a}[A_t^a] + \frac{1}{d_1} > \frac{1}{d_2} \tag{4.5.33}$$

is a sufficient condition for the existence of a positive real number s_1 such that $\Lambda_2(s_2) < 0$. Note, this condition means that the mean arrival rate at node 2 (away from Δ) is larger than the service rate. Evaluating Λ_2 at $s = 0$, we obtain

$$\begin{aligned}
\Lambda_2(0) &= \sum_{a\in A_2} -\alpha_a^*(0) + \sum_{a\in A_1} -\alpha_a(\theta) - \beta_2^*(0) - \beta_1^*(0) \\
&= \sum_{a\in A_2} -\alpha_a(\theta) + \sum_{a\in A_1} -\alpha_a(\theta) - \beta_2(-\theta) - \beta_1(-\theta) \qquad (4.5.34) \\
&= \Lambda(\theta) = 0.
\end{aligned}$$

The equality (4.5.34) follows from the definition of $\alpha_a^*(s)$ and $\beta_b^*(s)$ found in Lemma 3.5.1 and Lemma 3.6.2, respectively.

By Lemma 3.5.1 and Lemma 3.6.2, the derivative of Λ_2 at $s = 0$ is equal to

$$\Lambda_2'(0) = \sum_{a\in A_2} -E_{\varphi_a}[A_t^a] - \frac{1}{d_1} + \frac{1}{d_2}.$$

Under the inequality (4.5.33), this derivative is negative. Therefore, there exists a positive real number s_2 such that $\Lambda_2(s_2) < 0$.

Remark: The Lemma 4.5.1 and Lemma 4.5.4 follow directly from Lemma 4.5.6 and Lemma 4.5.7.

Chapter 5

Importance Sampling

5.1 Introduction

Importance sampling is a *speed up* (variance reduction) technique which can be used to efficiently estimate performance measures related to rare events in communication networks. We propose an importance sampling estimator to estimate the probability of extreme cell delays through an ATM-type network. For a fixed virtual connection, we want to estimate the long-run proportion of cells with an end-to-end cell transfer delay greater than the *large* fixed threshold maxCTD. We utilize the following notation :

$$D_\ell = \{ \text{ end-to-end CTD } > \text{ maxCTD } = \ell \}, \quad \alpha_\ell = \mathsf{P}_{\breve{\pi}}(D_\ell).$$

where $\breve{\pi}$ denotes the stationary distribution of the system. Note, in this chapter we consider the finite buffer network.

We present an importance sampling estimator for the probability of extreme cell delays through an ATM-type network.

5.2 The $\mathcal{A}$-cycle Method

We use the *twin* $\mathcal{A}$-cycle method, see [CHJS94, Heid95] to construct a strongly consistent estimator for the probability α_ℓ of extreme cell delays.

The *twin* $\mathcal{A}$-cycle method exploits the fact that the total encumbrance driven by the twisted sources has positive drift. A *twin* $\mathcal{A}$-cycle consists of two $\mathcal{A}$-cycles (the biased $\mathcal{A}$-cycle and the unbiased $\mathcal{A}$-cycle), both starting from the same state in $\mathcal{A}$, where $\mathcal{A}$ is a subset of the state space. The biased $\mathcal{A}$-cycle uses the twisted transition probabilities (see Section 5.3). Once started, a biased $\mathcal{A}$-cycle continues until the process hits $\mathcal{A}$ again. If the total encumbrance reaches (or exceeds) the critical value ℓ (the chosen quantile maxCTD), then we "untwist" the sources, i.e. return to the original transition probabilities. Note that we turn off the twist when the total encumbrance exceeds ℓ and not when an extreme cell delay is observed, so in fact even if the total encumbrance overloads during a biased $\mathcal{A}$-cycle we may not observe an extreme cell delay. Together with a biased $\mathcal{A}$-cycle we simulate an unbiased $\mathcal{A}$-cycle, i.e. with the original probabilities. Each return to a state in $\mathcal{A}$ ends an $\mathcal{A}$-cycle and generates a new initial state for the next pair of unbiased and biased $\mathcal{A}$-cycles. By forcing a biased $\mathcal{A}$-cycle to start at the return state of an unbiased $\mathcal{A}$-cycle we essentially start the twisted process off in the steady state $\breve{\pi}$ of the original process on $\mathcal{A}$.

The probability of extreme cell delays, α_ℓ, for the tagged cell stream is the long-run number of tagged cells with an end-to-end CTD greater than maxCTD (per $\mathcal{A}$-cycle) divided by the long-run number of tagged cells served at the last (most downstream) node in the network (per $\mathcal{A}$-cycle). We can estimate the denominator of this ratio using crude Monte-Carlo simulation techniques. Let $\mathrm{NCS}_{\mathcal{A}}[n]$ be the number of tagged cells served at the most downstream node during the nth unbiased $\mathcal{A}$-cycle. Therefore $(1/N)\sum_{n=1}^{N}\mathrm{NCS}_{\mathcal{A}}[n]$ is a strongly consistent estimator of the denominator, i.e. the long-run number of tagged cells per $\mathcal{A}$.

The numerator of the above ratio cannot be efficiently estimated from untwisted cycles with crude Monte-Carlo simulations, since extreme cell delays are rare events. Instead we use the biased $\mathcal{A}$-cycle. To demonstrate this fact we use an argument found in [LC01]. The squared coefficient of variation (SCV) of the number of extreme cell

delays during an unbiased $\mathcal{A}$-cycle, $\mathrm{NED}_{\mathcal{A}}$, with respect to $\mathsf{P}_{\breve{\pi}}$ is

$$\begin{aligned}\frac{\mathrm{VAR}_{\breve{\pi}}(\mathrm{NED}_{\mathcal{A}})}{(\mathsf{E}_{\breve{\pi}}(\mathrm{NED}_{\mathcal{A}}))^2} &= \frac{\mathsf{E}_{\breve{\pi}}(\mathrm{NED}_{\mathcal{A}}^2) - (\mathsf{E}_{\breve{\pi}}(\mathrm{NED}_{\mathcal{A}}))^2}{(\mathsf{E}_{\breve{\pi}}(\mathrm{NED}_{\mathcal{A}}))^2} \\ &\geq \frac{\mathsf{E}_{\breve{\pi}}(\mathrm{NED}_{\mathcal{A}}) - (\mathsf{E}_{\breve{\pi}}(\mathrm{NED}_{\mathcal{A}}))^2}{(\mathsf{E}_{\breve{\pi}}(\mathrm{NED}_{\mathcal{A}}))^2} \\ &= \frac{1}{\mathsf{E}_{\breve{\pi}}(\mathrm{NED}_{\mathcal{A}})} - 1,\end{aligned}$$

where the inequality follows from the fact that $\mathrm{NED}_{\mathcal{A}}$ is an integer valued random variable. Thus, $\mathrm{SCV}_{\breve{\pi}}(\mathrm{NED}_{\mathcal{A}}) \to \infty$ as $\mathsf{E}_{\breve{\pi}}(\mathrm{NED}_{\mathcal{A}}) \to 0$.

Let $\mathrm{NED}_{\mathcal{A}}[n]$ be the number of tagged cells with an end-to-end delay greater than maxCTD during the nth biased $\mathcal{A}$-cycle. To compensate for the twist, we must multiply $\mathrm{NED}_{\mathcal{A}}[n]$ by an importance sampling weight (likelihood ratio) $L[n]$ to get a strongly consistent estimator for the mean number of extreme cell delays observed per $\mathcal{A}$-cycle:

$$\frac{1}{N}\sum_{n=1}^{N} \mathrm{NED}_{\mathcal{A}}[n]\, L[n].$$

Similar to calculations from Appendix B in [DLM00] we can easily show that the likelihood ratio $L[n]$ is the product of two factors: the partial likelihood factor $\mathcal{L}_1[n]$ and a correction factor $\mathcal{L}_2[n]$, see Section 5.4.

For the nth biased Δ-cycle, let w_0 and $w_{\ell *}$ respectively denote the initial state of the system and the state of the system when the total encumbrance hits or exceeds the critical value ℓ for the first time. We define the partial likelihood ratio as

$$\mathcal{L}_1[n] := h(w_0)/h(w_{\ell *})$$

where h was defined in Theorem 4.2.1

To summarize, we present our importance sampling algorithm.

Algorithm 5.2.1 (The *Twin* $\mathcal{A}$-Cycle)

1. *Set* $r = 1$.

2. *Start the replication at an initial state in* $\mathcal{A}$. *Set* $n = 1$.

3. *Run the twisted process. Count the lost encumbrance $C[n]$ due to cell loss.*

4. *If the total encumbrance reaches (or exceeds) ℓ before hitting $\mathcal{A}$, go to step (5). If the total encumbrance returns to $\mathcal{A}$ first, go to step (6).*

5. *Calculate $\mathcal{L}_1[n]$ and $\mathcal{L}_2[n]$, see Section 5.4. Untwist the sources and continue to run the process until we return to $\mathcal{A}$, and count the number of tagged cells with an end-to-end cell CTD greater than maxCTD, $NED_{\mathcal{A}}[n]$. Go to step (7).*

6. *Stop the twisted process. No extreme cell delays were observed. We set $NED_{\mathcal{A}}[n] = 0$. Go to step (7).*

7. *Return the system to the state it was in before the last twisted run. Run the untwisted process until it again returns to $\mathcal{A}$. Count $NCS_{\mathcal{A}}[n]$, the number of tagged cells served at the most downstream node during this unbiased $\mathcal{A}$-cycle.*

8. *Update the initial state to be the return state of the untwisted process just generated. Set n=n+1.*

9. *Repeat steps (3) to (8) for N $\mathcal{A}$-cycles. Go to step (10).*

10. *Set*

$$Y_r = \left(\sum_{n=1}^{N} NDE_{\mathcal{A}}[n]\, \mathcal{L}_1[n]\, \mathcal{L}_2[n] \right) \quad \text{and} \quad X_r = \left(\sum_{n=1}^{N} NCS_{\mathcal{A}}[n] \right).$$

Set $r = r + 1$. Go to step (11).

11. *Repeat steps (2) to (10) for R independent replications. Go to step (12).*

12. *Our importance sampling estimator is the following ratio estimator*

$$\widehat{\alpha_\ell} = \overline{Y} \,/ \overline{X}\,.$$

The variance estimator is

$$\widehat{\sigma^2} = \frac{S_Y^2 + (\widehat{\alpha_\ell})^2\, S_X^2 - 2\,\widehat{\alpha_\ell}\, S_{XY}}{R\,(\overline{X})^2},$$

where $\overline{Y}, \overline{X}, S_X^2, S_Y^2$ and S_{XY} are the sample means, the sample variances and the sample covariance of the $\{Y_r\}$ and $\{X_r\}$, respectively. The relative error for a 99% confidence interval is given by

$$RE = 2.57\,\widehat{\sigma}\,/\widehat{\alpha_\ell}\,.$$

Remark 5.2.2

1. *In step (2) of Algorithm 5.2.1, we would like to generate the initial state in $\mathcal{A}$ from the distribution $\breve{\pi}_{\mathcal{A}}(\cdot) := \breve{\pi}(\cdot)/\breve{\pi}(\mathcal{A})$. Usually this is done approximately using a burn-in technique. We simulate M (a large number) returns to $\mathcal{A}$ and utilize the Mth return to $\mathcal{A}$ as the initial state.*

2. *We present an alternative to the burn-in technique in Chapter 7 using Perfect Sampling.*

3. *In Chapter 6, we investigate different variants of the Algorithm 5.2.1. The $\mathcal{A}$-cycles are either defined using returns to Δ or $\mathcal{G}_\Delta$, similar to [DLM00], or as in [LC96, LC01, FDL99], we also consider $\mathcal{A}$-cycles as busy periods, that is the set $\mathcal{A}$ represents the states for which the buffers at tagged nodes are all empty.*

5.3 Change of Measure and Modified Encumbrance

We would like to overload all of the tagged nodes. We propose an algorithm which modifies the set of tagged nodes. If, at any of the tagged nodes, the mean arrival cell rate is less than the service rate (under the twisted rates), then expunge this node from the set of the tagged nodes. Subsequent to this change, redefine the cross traffic C_{vc} and recalculate the twist. Below we explain how to accomplish this task.

We will define the following vectors $(\lambda(a,b))_{A\times B}$, and $(\mu(a,b))_{A\times B}$, called the controlled input and output rates, respectively. They will be defined recursively.

We utilize the following notation :

- We denote the levels of the ATM-type network with the symbol l.

- The collection of nodes on level l is denoted $B[l]$.
- The collection of sources that offer cells directly to node b is denoted A_b.
- The collection of sources that offer cells that pass through node b to reach their destination is denoted $A(b)$.
- The collection of nodes that offer cells directly to node b is denoted $B(b)$.

Definition 5.3.1 (Controlled Rates)

- *Let $l = 1$. For each $b \in B[1]$ and $a \in A_b$, define the controlled input as*

$$\lambda(a,b) := E_{\varphi_a}[A_t^a],$$

as the mean number of cells offered by source a per time slot under the twist. The controlled output is defined as follows. If

$$\sum_{a \in A_b} \lambda(a,b) < 1/d_b,$$

then $\mu(a,b) := \lambda(a,b)$. Otherwise,

$$\mu(a,b) := \frac{1}{d_b}\left(\frac{\lambda(a,b)}{\sum_{a' \in A_b}\lambda(a',b)}\right).$$

- *Let $l = n > 1$. For each $b \in B[n]$, define the controlled inputs as follows :*

  1. *For each $a \in A_b$, then*

$$\lambda(a,b) := E_{\varphi_a}[A_t^a].$$

  2. *For each $b' \in B(b)$, and $a \in A(b) \cup A(b')$, define*

$$\lambda(a,b) := \mu(a,b').$$

For each $b \in B[n]$, and $a \in A(b)$, the controlled outputs are defined as follows. If

$$\sum_{a \in A_b} \lambda(a,b) < 1/d_b,$$

then $\mu(a,b) := \lambda(a,b)$. *Otherwise,*

$$\mu(a,b) := \frac{1}{d_b}\left(\frac{\lambda(a,b)}{\sum_{a' \in A_b}\lambda(a',b)}\right).$$

- *For each* $b \in B$, *if* $a \notin A(b)$, *then* $\lambda(a,b) = \mu(a,b) = 0$.

Remark 5.3.2

1. *In the case of ATM-type intree networks, the stationary mean number of cells offered to buffer b, under the twist, is given by*

$$\sum_{a \in A}\lambda(a,b),$$

see [CHJS94]. This allows us to check Condition C.4 for such networks.

2. *It should be noted that in general, output streams do not necessarily decouple, see [GO98]. Nonetheless, we believe that these controlled input-output relations give the practical user a method to heuristically verify for the stability-instability of tagged node under the twist. The techniques found in [DM97] could possibly be adapted to the ATM-type network setting to better answer this question.*

We now present our proposed algorithm to compute the change of measure for our IS-based estimator.

Algorithm 5.3.3 (Change of Measure)

1. *Find the unique positive real number* θ *that satisfies the equation* $\Lambda(\theta) = 0$;

2. *For all sources that contribute to the encumbrance, i.e.* $a \in T_{vc} \cup C_{vc}$, *twist the kernel* $\hat{K}^a$ *for the underlying Markov chain* $\hat{M}^a$, *that is*

$$\hat{K}^a_\theta(m,m') = \widehat{K}^a_M \frac{r_a(m')}{r_a(m)},$$

see Section 3.3.

3. *With these twisted probabilities obtain the controlled input vector* $(\lambda(a,b))_{A\times B}$.

4. *Consider a tagged node $b \in TN$, verify that the controlled input for b is greater than the service rate, that is*

$$\sum_{a \in A} \lambda(a, b) > \frac{1}{d_b}.$$

If the inequality is not satisfied, then set $TN = TN - \{b\}$. Repeat for all $b \in TN$.

5. *If the set of tagged nodes TN has remained the same after step (4), then stop. Otherwise, redefine the cross traffic C_{vc} according to the tagged nodes and continue to step (1).*

5.4 Likelihood Ratio

In this section, we give the form of the likelihood ratio L. The functions $\mathcal{L}_1$ and $\mathcal{L}_2$, for Algorithm 5.2.1, are given at the end of this section. Denote the state space for the $\mathcal{A}$-cycles as $\Omega_{\mathcal{A}}$. The probability masses function for the biased and the unbiased $\mathcal{A}$-cycles are $\widetilde{P}_{\vec{\pi}}$ and $P_{\vec{\pi}}$, respectively.

Let $\omega \in \Omega_{\mathcal{A}}$ be an $\mathcal{A}$-cycle, $NED_{\mathcal{A}}(\omega)$ will be the number of extreme cell delays observed during this $\mathcal{A}$-cycle. The importance sampling weight function (or likelihood ratio) $L(\omega)$ must satisfy

$$\sum_{w \in \Omega_{\mathcal{A}}} \widetilde{P}_{\vec{\pi}}(\omega)\, L(\omega)\, \mathrm{NED}_{\mathcal{A}}(\omega) = \sum_{w \in \Omega_{\mathcal{A}}} P_{\vec{\pi}}(\omega) \mathrm{NED}_{\mathcal{A}}(\omega).$$

Recall that the underlying Markov chain $\hat{M}^a$ has the transition kernel $\hat{K}^a$. We saw in Section 4.2.2, that under the twist, we replace the transition kernel $\hat{K}^a$ with the kernel $\hat{K}^a_\theta$, which is defined in Section 3.3, if source a contributes to the encumbrance, i.e. $a \in T_{vc} \cup C_{vc}$. If the source does not contribute to the encumbrance, i.e. $a \in R_{vc}$, then even under the twist the kernel $\hat{K}^a_\theta$ remains unchanged.

Let $\omega = (w_0, \ldots, w_k, w_{k+1}, \ldots, w_{k*})$ be a biased $\mathcal{A}$-cycle for which the twisted probabilities are used for the first k transitions. The transitions of the underlying Markov chains do not occur at every transition of the global process. Let $k(a)$ denote

the number of transitions for the underlying Markov chain for source a using the twisted kernel $\hat{K}^a_\theta$. The likelihood ratio will have the following form,

$$\begin{aligned} L(\omega) = \frac{P_{\breve{\pi}}(\omega)}{\widetilde{P}_{\breve{\pi}}(\omega)} &= \prod_{a \in T_{vc} \cup C_{vc}} \frac{K^a(m^a_0, m^a_1) \cdots K^a(m^a_{k(a)-1}, m^a_{k(a)})}{\hat{K}^a_\theta(m^a_0, m^a_1) \cdots \hat{K}^a_\theta(m^a_{k(a)-1}, m^a_{k(a)})} \\ &= \prod_{a \in T_{vc} \cup C_{vc}} \frac{K^a(m^a_0, m^a_1) \cdots K^a(m^a_{k(a)-1}, m^a_{k(a)})}{\widehat{K}^a_M(m^a_0, m^a_1) \frac{r_a(m^a_1)}{r_a(m^a_0)} \cdots \widehat{K}^a_M(m^a_{k(a)-1}, m^a_{k(a)}) \frac{r_a(m^a_{k(a)-1})}{r_a(m^a_{k(a)})}} \\ &= \prod_{a \in T_{vc} \cup C_{vc}} \frac{K^a(m^a_0, m^a_1) \cdots K^a(m^a_{k(a)-1}, m^a_{k(a)})}{\widehat{K}^a_M(m^a_0, m^a_1) \cdots \widehat{K}^a_M(m^a_{k(a)-1}, m^a_{k(a)})} \cdot \frac{r_a(m^a_0)}{r_a(m^a_{k(a)})}, \end{aligned}$$

where the last equality follows from telescoping.

By definition,

$$\widehat{K}^a_M(m, m') = e^{[\alpha_a c_a(m') + \varepsilon_a \theta \iota_a(m)]} \hat{K}^a(m, m').$$

Hence,

$$L(\omega) = \prod_{a \in T_{vc} \cup C_{vc}} \exp\left\{ \sum_{i=1}^{k(a)} -\alpha_a c_a(m^a_i) - \varepsilon_a \theta \iota_a(m^a_{i-1}) \right\} \frac{r_a(m^a_0)}{r_a(m^a_{k(a)})}.$$

Recall, $c_a(m)$ represents sojourn time in state m. Thus,

$$k = u^a_0 - u^a_k + \sum_{i=1}^{k(a)} c_a(m^a_i),$$

where u^a_0 and u^a_k represent the residual number of time slots until the next transition of the underlying Markov chain for source a, initially and after k transitions of the global process, respectively. Thus,

$$L(\omega) = \prod_{a \in T_{vc} \cup C_{vc}} e^{-\alpha_a k} \exp\left\{ \sum_{i=1}^{k(a)} -\varepsilon_a \theta \iota_a(m^a_{i-1}) \right\} \frac{e^{\alpha_a(u_0)} r_a(m^a_0)}{e^{\alpha_a(u_k)} r_a(m^a_{k(a)})}.$$

Consider a tagged node $b \in TN$. From Section 3.6, the function β_b evaluated at $-\theta$ is equal to

$$\beta_b(-\theta) = \theta.$$

We also know that the residual number of time slots until the next service slot, v_t^b, satisfies, for all $t \in \mathbb{Z}_+$,

$$v_{t+1}^b - v_t^b = -1 + d_b\, \mathbf{1}(v_t^b = 0).$$

Thus, for $b \in TN$,

$$\begin{aligned}
& e^{-k\,\beta_b(-\theta)}\, \frac{e^{\beta_b(-\theta)\, v_0^b}}{e^{\beta_b(-\theta)\, v_k^b}}\, \exp\left\{\theta\, d_b \sum_{t=0}^{k-1} 1(v_t^b = 0)\right\} \\
= \;& e^{-k\,\beta_b(-\theta)} \prod_{t=0}^{k-1} e^{\theta\, d_b\, 1(v_t^b=0)}\, \frac{e^{\beta_b(-\theta)\, v_t^b}}{e^{\beta_b(-\theta)\, v_{t+1}^b}} \quad \text{(by telescoping)} \\
= \;& e^{-k\,\beta_b(-\theta)} \prod_{t=0}^{k-1} e^{\theta\, d_b\, 1(v_t^b=0)}\, e^{\beta_b(-\theta)\,(1-d_b\,\mathbf{1}(v_t^b=0))} \quad (\text{since } v_{t+1}^b - v_t^b = -1 + d_b\, \mathbf{1}_{(v_t^b=0)}) \\
= \;& 1 \qquad (\text{since } \beta_b(-\theta) = \theta).
\end{aligned}$$

Thus,

$$\begin{aligned}
L(\omega) \;=\; & \prod_{a \in T_{vc} \cup C_{vc}} e^{-\alpha_a\, k}\, \exp\left\{\sum_{i=1}^{k(a)} -\varepsilon_a\, \theta\, \iota_a(m_{i-1}^a)\right\}\, \frac{e^{\alpha_a(u_0)}\, r_a(m_0^a)}{e^{\alpha_a(u_k)}\, r_a(m_{k(a)}^a)} \\
& \times \prod_{b \in TN} e^{-k\,\beta_b(-\theta)}\, \frac{e^{\beta_b(-\theta)\, v_0^b}}{e^{\beta_b(-\theta)\, v_k^b}}\, \exp\left\{\theta\, d_b \sum_{t=0}^{k-1} 1(v_t^b = 0)\right\} \\
= \;& \exp\left\{k \left(-\sum_{a \in T_{vc} \cup C_{vc}} \alpha_a(\theta) - \sum_{b \in TN} \beta_b(-\theta)\right)\right\} \\
& \times \exp\left\{-\theta \left(\sum_{a \in T_{vc} \cup C_{vc}} \sum_{i=1}^{k(a)} \varepsilon_a\, \iota_a(m_{i-1}^a) - \sum_{b \in TN} \sum_{t=0}^{k-1} d_b\, \mathbf{1}(v_t = 0)\right)\right\} \\
& \times \prod_{a \in T_{vc} \cup C_{vc}} \frac{e^{\alpha_a(u_0)}\, r_a(m_0^a)}{e^{\alpha_a(u_k)}\, r_a(m_{k(a)}^a)} \times \prod_{b \in TN} \frac{e^{\beta_b(-\theta)\, v_0^b}}{e^{\beta_b(-\theta)\, v_k^b}}
\end{aligned}$$

We simplify the likelihood ratio using $\hat{a}_\theta(\hat{w})$ and the fact that

$$0 = \Lambda(\theta) = -\sum_{a \in T_{vc} \cup C_{vc}} \alpha_a(\theta) - \sum_{b \in TN} \beta_b(-\theta),$$

$$L(\omega) = \frac{\hat{a}_\theta(\hat{w}_0)}{\hat{a}_\theta(\hat{w}_k)}\, \exp\left\{-\theta \left(\sum_{a \in T_{vc} \cup C_{vc}} \sum_{i=1}^{k(a)} \varepsilon_a\, \iota_a(m_{i-1}^a) - \sum_{b \in TN} \sum_{t=0}^{k-1} d_b\, \mathbf{1}(v_t = 0)\right)\right\}.$$

We finish by relating the sum in the latter exponential to the difference in the total encumbrance $e_k - e_0$. The difference $e_k - e_0$ is equal to the encumbrance due to the arrivals minus the encumbrance due to the services, as long as there are no cell losses due to finite buffers or lost services due to empty buffers at a tagged node.

We discussed the case of the lost services in Section 4.4. Therein, we had defined the lost services function

$$\zeta_\Delta(w) = \sum_{b \in O_B(w)} d_b.$$

The latter represents the extra amount of encumbrance that would have been decremented in a transition from w to w', if all the tagged nodes were non-empty.

Suppose that between the ends of the time slots t and $t+1$, one (or more cells) was lost due to a finite buffer. Let c represent a cell. We will assume that it was lost at level l. We will denote the set of remaining nodes through which it should have been buffered and served as $TN(c,l)$, where l represents the level at which it was lost. Since the cell was lost we must decrement the total encumbrance from t to $t+1$ by

$$\sum_{b \in TN(c,l)} d_b.$$

We will denote the sum of these values over the $\mathcal{A}$-cycle ω, while the process is twisted, as $C(\omega)$.

Thus, the difference in total encumbrance $e_k - e_0$ is equal to

$$\sum_{t=0}^{k-1} \zeta_\Delta(w_t) + \left(\sum_{a \in T_{vc} \cup C_{vc}} \sum_{i=1}^{k(a)} \varepsilon_a \, \iota_a(m_{i-1}^a) - \sum_{b \in TN} \sum_{t=0}^{k-1} d_b \, \mathbf{1}(v_t = 0) \right) - C(\omega).$$

Therefore, the likelihood ratio is equal to

$$\begin{aligned} L(\omega) &= \frac{e^{\theta e_0} \, \hat{a}_\theta(\hat{w}_0)}{e^{\theta w_k} \, \hat{a}_\theta(\hat{w}_k)} \, e^{\theta C(\omega)} \, \exp\left\{ -\theta \sum_{t=0}^{k-1} \zeta_\Delta(w) \right\} \\ &= \frac{h(w_0)}{h(w_k)} \, e^{\theta C(\omega)} \, \exp\left\{ -\theta \sum_{t=0}^{k-1} \zeta_\Delta(w) \right\} \end{aligned}$$

We call $\mathcal{L}_1(\omega) := h(w_0)/h(w_k)$ the partial likelihood ratio and

$$\mathcal{L}_2(\omega) := e^{\theta C(\omega)} \, \exp\left\{ -\theta \sum_{t=0}^{k-1} \zeta_\Delta(w) \right\}$$

the correction factor.

We summarize the result in the following theorem.

Theorem 5.4.1 (Likelihood Ratio) *Let $\omega = (w_0, \ldots, w_k, w_{k+1}, \ldots, w_{k*})$ be a biased $\mathcal{A}$-cycle for which the twisted probabilities are used for the first k transitions. Its likelihood ratio is $L(\omega) = \mathcal{L}_1(\omega) \times \mathcal{L}_2(\omega)$, where*

$$\mathcal{L}_1(\omega) = h(w_0)/h(w_k),$$

$$\mathcal{L}_2(\omega) := e^{\theta C(\omega)} \exp\left\{-\theta \sum_{t=0}^{k-1} \zeta_\Delta(w)\right\},$$

and $C(\omega)$ represents the encumbrance loss due to cell loss, while the process was twisted.

5.5 Consistency

We end the chapter by proving that our importance sampling estimator is consistent. Let $\breve{W}$ be the delay process under the finite buffers regime. We will assume that it is an ergodic Markov chain with stationary distribution $\breve{\pi}$, transition kernel $\breve{K}$ and state space S. We will need to distinguish between frequencies per time slot and frequencies per $\mathcal{A}$-cycle. To do so we introduce the following notation: NC_t and NED_t represent the number of tagged cells served during the time slot t and the number of observed extreme cell delays during the time slot t, respectively. Their $\mathcal{A}$-cycle counterparts are $NC_{\mathcal{A}}[n]$ and $NED_{\mathcal{A}}[n]$ (or $\mathcal{NED}_{\mathcal{A}}[n]$ in the case of a twisted $\mathcal{A}$-cycle), respectively.

We will also need the following counting processes :

$$N(t) = \ \# \text{ of } \mathcal{A}\text{-cycles by time slot } t$$

and

$$N_w(t) = \ \# \text{ of returns to state } w \text{ by time slot } t.$$

In order to condition on the initial state of an $\mathcal{A}$-cycle we introduce the following notation:

$$NC^w_{\mathcal{A}}[k] = k\text{th } \mathcal{A}\text{-cycle with initial state } w.$$

We will also need the following stopping times,

$$\tau_w = \inf\{t > 0 \ : \ W_t = w\}, \qquad \tau_{\mathcal{A}}^w[0] = \inf\{t > 0 \ : \ W_t \in \mathcal{A}, W_0 = w\},$$

and

$$\tau_{\mathcal{A}}^w[k] = \inf\{t > \tau_{\mathcal{A}}^w[k-1] \ : \ W_t \in \mathcal{A}, W_0 = w\}, \text{ for } k = 1, 2, 3, \ldots$$

Define the watched (censored) process $J_{\mathcal{A}}[k]$, where $J_{\mathcal{A}}[0] = W_0$, where $W_0 \in \mathcal{A}$ and

$$J_{\mathcal{A}}[k] = W_{\tau_{\mathcal{A}}[k-1]}, \quad \text{for } k = 1, 2, \ldots$$

From renewal theory, the probability of extreme cell delays is,

$$\alpha_\ell = \lim_{T\to\infty} \frac{\sum_{t=0}^{T} NED_t/T}{\sum_{t=0}^{T} NC_t/T} = \frac{\mathsf{E}_w\left[\sum_{t=0}^{\tau_w - 1} NED_t\right] / \mathsf{E}_w\left[\tau_w\right]}{\mathsf{E}_w\left[\sum_{t=0}^{\tau_w - 1} NC_t\right] / \mathsf{E}_w\left[\tau_w\right]} \qquad \text{(almost surely)},$$

where $w \in S$ and τ_w is the first return time to w. We now show that if we define the cycles as returns to some set $\mathcal{A}$, then we still obtain the same limit.

Theorem 5.5.1 (Consistency) *Under the assumption that $\breve{W}$ is an ergodic Markov chain, then the importance sampling estimator is strongly consistent, that is*

$$\frac{\sum_{n=1}^{N(t)} \mathcal{NED}_{\mathcal{A}}[n]\, L_n}{\sum_{n=1}^{N(t)} NC_{\mathcal{A}}[n]} \longrightarrow \alpha_\ell$$

almost surely as $t \to \infty$.

Proof : First observe that for a fixed $w \in S$, the random variables in the sequence $\{NC_{\mathcal{A}}^w[k]\}_{k=1}^{\infty}$ are independent and identically distributed, thus by the strong law of large numbers

$$\frac{1}{N_w(t)} \sum_{k=1}^{N_w(t)} NC_{\mathcal{A}}^w[k] \longrightarrow E_w\left[NC_{\mathcal{A}}[1]\right],$$

almost surely as $t \to \infty$. It is also well known that

$$\frac{N_w(t)}{t} \to \breve{\pi}(w) \qquad \text{and} \qquad \frac{N(t)}{t} \to \breve{\pi}(\mathcal{A}),$$

almost surely as $t \longrightarrow \infty$. Combining these three limits we obtain the following result:

$$\begin{aligned}\frac{1}{N(t)} \sum_{n=1}^{N(t)} NC_{\mathcal{A}}[n] &= \sum_{w \in \mathcal{A}} \left[\left(\frac{N_w(t)}{t} \right) \left(\frac{t}{N(t)} \right) \left(\frac{1}{N_w(t)} \sum_{k=1}^{N_w(t)} NC_{\mathcal{A}}^w[k] \right) \right] \\ &\longrightarrow \sum_{w \in \mathcal{A}} \breve{\pi}_{\mathcal{A}}(w)\, \mathsf{E}_w[NC_{\mathcal{A}}[1]],\end{aligned}$$

almost surely as $t \longrightarrow \infty$, where $\breve{\pi}_{\mathcal{A}}(w) = \breve{\pi}(w)/\breve{\pi}(\mathcal{A})$.

Using similar arguments we can easily show that

$$\frac{1}{N(t)} \sum_{n=1}^{N(t)} \mathcal{NED}_{\mathcal{A}}[n]\, L_n \longrightarrow \sum_{w \in \mathcal{A}} \breve{\pi}_{\mathcal{A}}(w)\, \widetilde{\mathsf{E}}_w[\mathcal{NED}_{\mathcal{A}}[1]\, L_1],$$

almost surely as $t \longrightarrow \infty$, where $\widetilde{\mathsf{E}}_w$ is the mean operator for the biased (twisted) measure $\widetilde{\mathsf{P}}_w$. In Section 5.4, we showed that

$$\widetilde{\mathsf{E}}_w[\mathcal{NED}_{\mathcal{A}}[1]\, L_1] = \mathsf{E}_w[NED_{\mathcal{A}}[1]]$$

then

$$\frac{1}{N(t)} \sum_{n=1}^{N(t)} NED_{\mathcal{A}}[n]\, L_n \longrightarrow \sum_{w \in \mathcal{A}} \breve{\pi}_{\mathcal{A}}(w)\, \mathsf{E}_w[NED_{\mathcal{A}}[1]],$$

almost surely as $t \longrightarrow \infty$.

It remains to show that

$$\frac{\sum_{w \in \mathcal{A}} \breve{\pi}_{\mathcal{A}}(w)\, \mathsf{E}_w[NED_{\mathcal{A}}[1]]}{\sum_{w \in \mathcal{A}} \breve{\pi}_{\mathcal{A}}(w)\, \mathsf{E}_w[NC_{\mathcal{A}}[1]]} = \frac{\mathsf{E}_w\left[\sum_{t=0}^{\tau_w - 1} NED_t\right]}{\mathsf{E}_w\left[\sum_{t=0}^{\tau_w - 1} NC_t\right]} = \frac{\breve{\pi}(w)\, \mathsf{E}_w\left[\sum_{t=0}^{\tau_w - 1} NED_t\right]}{\breve{\pi}(w)\, \mathsf{E}_w\left[\sum_{t=0}^{\tau_w - 1} NC_t\right]}.$$

To do so, we will show that

$$\sum_{w \in \mathcal{A}} \breve{\pi}_{\mathcal{A}}(w)\, \mathsf{E}_w[NC_{\mathcal{A}}[1]] = \breve{\pi}(w)\, \mathsf{E}_w\left[\sum_{t=0}^{\tau_w - 1} NC_t\right].$$

We can prove a similar result for the numerator using the same arguments as below.

Observe that by the strong Markov property we obtain

$$\begin{aligned}&\mathsf{E}_w\left[\sum_{t=\tau_{\mathcal{A}}^w[0]+\tau_{\mathcal{A}}^w[1]+\ldots+\tau_{\mathcal{A}}^w[k-1]}^{\tau_{\mathcal{A}}^w[0]+\tau_{\mathcal{A}}^w[1]+\ldots+\tau_{\mathcal{A}}^w[k]} NC_t \,\middle|\, \tau_{\mathcal{A}}[0], \ldots, \tau_{\mathcal{A}}[k-1], J_{\mathcal{A}}[0], \ldots, J_{\mathcal{A}}[k-1]\right] \\ = \;& \mathsf{E}_{w'}\left[\sum_{t=0}^{\tau_{\mathcal{A}}[0]-1} NC[t]\right], \quad \text{on } \{J_{\mathcal{A}}[k-1] = w'\}.\end{aligned}$$

Thus,

$$\mathsf{E}_w\left[\sum_{t=0}^{\tau_w-1} NC_t\right] = \mathsf{E}_w\left[\sum_{k=0}^{\tau-1} \mathsf{E}_{J_{\mathcal{A}}[k]}\left[\sum_{t=0}^{\tau_{\mathcal{A}}^{J_{\mathcal{A}}[k]}[0]} NC_t\right]\right],$$

where $\tau = \inf\{k > 0 \ : \ J_{\mathcal{A}}[k] = w\}$. As we condition on the watched chain, we observe that this quantity is equal to

$$\begin{aligned}
&\sum_{w' \in \mathcal{A}} \mathsf{E}_{w'}\left[\sum_{t=0}^{\tau_{\mathcal{A}}^{w'}[0]-1} NC_t\right] \mathsf{E}_w\left[\sum_{k=0}^{\tau-1} \mathbf{1}(J_{\mathcal{A}}[k] = w')\right] \\
= \ & \mathsf{E}_w[\tau] \sum_{w' \in \mathcal{A}} \mathsf{E}_{w'}\left[\sum_{t=0}^{\tau_{\mathcal{A}}^{w'}[0]-1} NC_t\right] \mathsf{E}_w\left[\sum_{k=0}^{\tau-1} \mathbf{1}(J_{\mathcal{A}}[k] = w')\right] \Big/ \mathsf{E}_w[\tau] \\
= \ & \mathsf{E}_w[\tau] \sum_{w' \in \mathcal{A}} \mathsf{E}_{w'}\left[\sum_{t=0}^{\tau_{\mathcal{A}}^{w'}[0]-1} NC_t\right] \breve{\pi}(w').
\end{aligned}$$

The latter equality follows from a well known characterization of the stationary distribution of a Markov chain, that is

$$\breve{\pi}(w') = \frac{E_w[\#\text{ of visits to } w' \text{ before returning to } w\ ]}{E_w[\tau_w].}$$

From the same characterization, we obtain $\breve{\pi}(w) = 1/\mathsf{E}_w[\tau]$. Thus,

$$\mathsf{E}_w\left[\sum_{t=0}^{\tau_w-1} NC_t\right] = \frac{1}{\breve{\pi}(w)} \sum_{w' \in \mathcal{A}} \breve{\pi}(w')\, \mathsf{E}_{w'}\left[\sum_{t=0}^{\tau_{\mathcal{A}}^{w'}[0]-1} NC_t\right].$$

By multiplying both sides by $\breve{\pi}(w)$ we obtain our desired result, that is

$$\breve{\pi}(w)\, \mathsf{E}_w\left[\sum_{t=0}^{\tau_w-1} NC_t\right] = \sum_{w' \in \mathcal{A}} \breve{\pi}(w')\, \mathsf{E}_{w'}\left[\sum_{t=0}^{\tau_{\mathcal{A}}^{w'}[0]-1} NC_t\right].$$

□

Chapter 6

Simulation Results

In this chapter, we estimate the probability of extreme cell delays for a particular virtual connection for a few specific models. Our goal is to compare the relative error (i.e. half width of the confidence interval) of the estimates obtained by a crude Monte-Carlo simulation to those obtained by using the $\mathcal{A}$-cycle Algorithm 5.2.1.

6.1 Two Node Intree Tandem Network

Figure 4 exhibits the network for the example. Fifteen independent bursty sources offer cells to the system. The sources are characterized by the following parameters: the constant inter-arrival time κ_a, the transition probability from *on* to *off* p^a_{10}, and the transition probability from *off* to *on* p^a_{01} for all $a \in A$.

The fifteen sources are partitioned into 4 classes. The cells in the classes 1 and 2

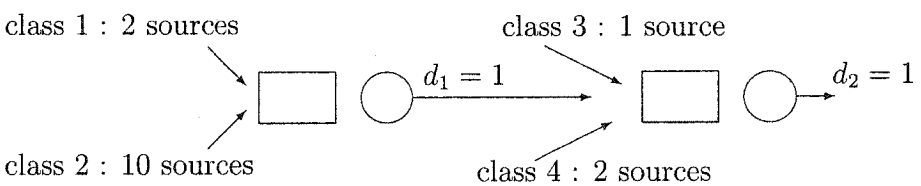

Figure 4: Intree Tandem Network : Two Nodes

offer cells to node 1. While the sources in the classes 3 and 4 offer cells to node 2. The cells leave the network from node 2.

The parameters for the sources in the classes 1 and 3 are

$$\kappa_a = 1 \text{ (time slot)}, \quad p^a_{01} = 0.01, \quad \text{and} \quad p^a_{10} = 0.1.$$

Thus, the mean idle period is $1/p^a_{01} = 100$ time slots and the mean burst size is $1/p^a_{10} = 10$ cells. Since, the constant inter-arrival time is $\kappa_a = 1$, then the mean burst period is 10 time slots. The mean number of cells offered to the system (per time slot) by such a source is

$$E_{\pi_a}[A^a_t] = \frac{1/p^a_{10}}{\kappa_a/p^a_{10} + 1/p^a_{01}} \approx 0.0909.$$

The parameters for the sources in the classes 2 and 4 are

$$\kappa_a = 2 \text{ (time slots)}, \quad p^a_{01} = 1/60, \quad \text{and} \quad p^a_{10} = 1/3.$$

Thus, the mean idle period is $1/p^a_{01} = 60$ time slots and the mean burst size is $1/p^a_{10} = 3$ cells. Since, the constant inter-arrival time is $\kappa_a = 2$, then the mean burst period is 6 time slots. The mean number of cells offered to the system (per time slot) by such a source is

$$E_{\pi_a}[A^a_t] = \frac{1/p^a_{10}}{\kappa_a/p^a_{10} + 1/p^a_{01}} \approx 0.04545.$$

The constant inter-service times at the nodes are respectively, $d_1 = 1$, $d_2 = 1$. Thus, the service rates at each of the nodes is 1 *cell per timeslot.* Each node has a buffer capacity of 1500 cells.

For the purpose of this example both nodes are tagged. We consider the tagged stream as the cell stream from a particular source in class 1. The other cell streams form the cross traffic. The cells that are offered to node 1 will increment the total encumbrance by $d_1 + d_2 = 2$, whereas the other cells will increment the total encumbrance by $d_2 = 1$.

We verify that the conditions of Theorem 4.2.5 are satisfied,

$$\sum_{a \in A} \varepsilon_a E_{\pi_a}[A^a_t] \approx 1.4544 < 2 = |TN|,$$

Class	p_{10}^a	$\hat{\mathcal{K}}_\theta^a(1,0)$	p_{01}^a	$\hat{\mathcal{K}}_\theta^a(1,0)$	$E_{\pi_a}[A_t^a]$	$E_{\varphi_a}[A_t^a]$
1	0.1	0.0466	0.01	0.0224	0.0909	0.3251
2	0.3333	0.2901	0.0167	0.0203	0.04545	0.0614
3	0.1	0.0718	0.01	0.0143	0.0909	0.1659
4	0.3333	0.3119	0.0167	0.0184	0.04545	0.0527

Table 1: Untwisted and Twisted Rates

and

$$\sum_{a\in A} \frac{\varepsilon_a}{d_A^a} - |TN| = 14 > 0.$$

Therefore, the harmonic function h does exist. The constant θ that satisfies $\Lambda(\theta) = 0$ is approximately equal to 0.03515. We can calculate the twisted probabilities from Section 3.3.1 with $\gamma = \theta$. The twisted rates are given in Table 1.

The mean drift for the queue size at nodes 1 and 2 (away from Δ) are both positive, i.e.

$$\sum_{a\in A_1} E_{\varphi_a}[A_t^a] - \frac{1}{d_1} \approx 0.2646 > 0,$$

and

$$\frac{1}{d_1} + \sum_{a\in A_2} E_{\varphi_a}[A_t^a] - 1/d_2 \approx 0.2712 > 0.$$

The conditions from Lemma 4.5.4 are satisfied. Thus, there exist ν_b, b=1,2, such that $\Lambda_b(\nu_b) < 0$, $b = 1, 2$. To satisfy the conditions of Theorem 4.5.5, we need to find $\nu_1 > d_1\theta \approx 0.03515$ and $\nu_2 > d_2\theta \approx 0.03515$ such that $\Lambda_b(\nu_b) < 0$, $b = 1, 2$. We evaluated Λ_1 at $\nu_1 = 0.04$ and Λ_2 at $\nu_2 = 0.04$, and obtained $\Lambda_1(0.04) \approx -0.000688 < 0$, and $\Lambda_2(0.04) \approx -0.0086 < 0$.

We can conclude that the large deviations for the total encumbrance is giving by Theorem 4.1.1, with $\theta \approx 0.03515$.

Simulation results comparing crude Monte-Carlo estimates, see Table 4, of the probability of extreme cell delays with those obtained from the Algorithm 5.2.1, with

$$\mathcal{A} = \Delta = \{w : q_1 = 0 \text{ or } q_2 = 0\},$$

are displayed in Table 2, or with

$$\mathcal{A} = \{w : q_1 = 0, q_2 = 0\},$$

are displayed in Table 3.

Ten independent replications were used to estimate the probability of extreme cell delays by $\widehat{PECD}_{\text{is}(1)}$, $\widehat{PECD}_{\text{is}(2)}$, or $\widehat{PECD}_{\text{cmc}}$. We compare the estimators by their estimated coefficient of variation, i.e. the estimated standard deviation over the estimated mean. Computational effort is measured in cells generated per replication.

The crude Monte Carlo estimates are found in Table 4. It is evident from the table that as the probability of extreme cell delays decreases the coefficient of variation or relative error increases for the same amount of computational effort. For a maxCTD of 300 time slots, we did not observe any extreme cell delays.

The importance sampling estimates when the $\mathcal{A}$-cycles are defined as returns to the boundary Δ are found in Table 3. Due to the work of [GK95] we decided to first investigate Algorithm 5.2.1 with $\mathcal{A} = \Delta$. In [GK95], the authors studied a two node Jackson network in tandem. They noticed that an importance sampling estimator based on large deviations results is not necessarily efficient and that it could even perform worse than a crude Monte Carlo estimator. This undesirable behaviour is mostly due the fact that the likelihood ratio is not well behaved on the boundary Δ. Thus, if we define the $\mathcal{A}$-cycles as returns to the boundary Δ, then the likelihood ratio should be "stable". Although, we have observed that even though the likelihood ratio was under control we could not reduce the coefficient of variation to a reasonable level. The extreme cell delays were still unlikely under the "twist". Or rather the system under the twist did produce extreme cell delays but the $\mathcal{A}$-cycles would often end before we could observe this extreme delay, i.e. the old cells would still be waiting to be served as one of the buffers emptied. To be able to observe these old cells leave the system we decided to extend the cycles.

The extended cycles are defined as returns to the origin, i.e.

$$\mathcal{A} = \{w : q_1 = 0, q_2 = 0\}.$$

By extending the $\mathcal{A}$-cycles in this way we allow the old cells, which were produced due to the overloading caused by the use of the twisted probabilities, enough time travel

through the network and leave. The results are in Table 2. We were able to obtain an estimate of a probability of extreme cell delay in the order of 10^{-8} with a coefficient of variation under 20% with an effort in the order of 10^5 cells per replication.

maxCTD	$\widehat{PECD}_{\text{is}(1)}$	*coeff. var.*	effort
100	1.49×10^{-2}	4.48%	7.63×10^3
200	2.42×10^{-4}	3.98%	7.32×10^4
250	2.23×10^{-5}	3.98%	7.32×10^4
350	3.72×10^{-7}	14.67%	1.14×10^5
400	4.66×10^{-8}	16.48%	4.31×10^5

Table 2: IS Estimations with returns to the origin

maxCTD	$\widehat{PECD}_{\text{is}(2)}$	*coeff. var.*	effort
100	1.20×10^{-2}	22.13%	3.63×10^3
200	1.17×10^{-4}	20.44%	2.18×10^5
250	1.72×10^{-5}	30.40%	6.48×10^5

Table 3: IS Estimations with returns to the edge Δ

maxCTD	$\widehat{PECD}_{\text{cmc}}$	*coeff. var.*	effort
100	1.52×10^{-2}	3.55%	4.10×10^6
200	2.20×10^{-4}	32.32%	4.10×10^6
250	1.35×10^{-5}	56.30%	4.10×10^6
300	0	???	4.10×10^6

Table 4: Crude Monte Carlo Estimations

Chapter 7

Perfect Sampling

7.1 Introduction

We present two variants of the Propp and Wilson [PW96] Coupling From the Past (CFTP) Algorithm. One replaces the partial ordering condition for *monotone* CFTP with a pre ordering condition. The other is a CFTP algorithm for a multi-dimensional Markov chain for which the marginal stationary distribution on a possibly denumerable state space is known.

Since the seminal paper of Propp and Wilson [PW96] there has been a considerable amount of work published on the development of exact sampling algorithms, i.e. algorithms that sample *perfectly* from the stationary distribution π in the context of Markov Chain Monte-Carlo. Many of these are adaptations of the Coupling From the Past (CFTP) Algorithm proposed by Propp and Wilson [PW96]. We present a variant of CFTP for a multi-dimensional Markov chain for which the marginal stationary distribution on a possibly denumerable state space is known.

In this chapter, we will present the adapted CFTP independently of the previous chapters. The distribution π is used in a general context and does not represent the stationary distribution for a process presented in the previous chapters. We will return to our ATM-type queueing and delay processes in the Sections 7.6 and 7.7.

The Propp-Wilson Algorithm allows the user to sample from the stationary distribution π at some fixed time, e.g. t=0. It works as follows. The user will simulate

independent processes starting at some particular time $T < 0$ in the past. There is a process for each state in the finite state space $\mathcal{S} = \{1, \ldots, n\}$. If two processes couple, then they continue as one unique process. If by time 0 the processes have not all coupled then we restart all the n processes at a time $T'(< T)$. Note, we must use the same pseudo-random number U_t at time t. If the processes do couple by time 0, then the output at time 0 is a sample from the distribution π.

The CFTP algorithm outlined in the preceding paragraph is very general in scope but not practical for large state spaces. Propp and Wilson [PW96] propose a CFTP algorithm for *Monotone* Markov Chains which is efficient even for very large state spaces. To apply this adapted algorithm there must be a partial ordering on the state space with unique maximal and minimal states. The probability transitions must also be order preserving. If these conditions are met, then Propp and Wilson [PW96] say that the Markov Chain gives a *monotone Monte Carlo algorithm* for approximating the stationary distribution π. Instead of running all n processes, we only need to run two processes, initialized at the maximal and minimal states. To use terminology from Corcoran and Tweedie [CT01], all other processes are *sandwiched* between the maximum and the minimum processes. This technique allows the user to identify when all the processes have coupled without the computational effort required to run all of the n processes.

We adapt the CFTP algorithm to a multi-dimensional setting. We consider the discrete time Markov chain with a possibly infinite state space $\mathcal{S} = \mathcal{S}_X \times \mathcal{S}_Y$, where $\mathcal{S}_X$ is a *denumerable* (or finite) state space and $\mathcal{S}_Y$ is a finite state space. The transition kernel, denoted K, is of the form

$$K(x, y; x', y') = K_X(x; x')\, K_Y(x, x', y; y'). \tag{7.1.1}$$

We will assume that K generates an irreducible, aperiodic, positive recurrent Markov Chain with stationary distribution π. We also assume that the user has *a priori* knowledge of the marginal distribution π_X, where

$$\pi_X(x) = \sum_{y \in \mathcal{S}_y} \pi(x, y), \quad \text{for all } x \in \mathcal{S}_X.$$

In Section 7.4, we adapt the Propp and Wilson algorithm for Monotone Markov

chains to our model. In this setting, we will require that the *monotonicity* properties be satisfied only on $\mathcal{S}_Y$ instead of the entire state space. In Section 7.6, we weaken the condition of partial ordering to a condition of pre ordering with a unique minimal (or maximal) state. We present the adapted algorithm in the context of our multi-dimensional setting. Similar modifications could be made to the Propp and Wilson algorithm for Monotone Markov chains.

In Section 7.6, we utilize the adapted CFTP algorithm to sample from the stationary distribution of queueing processes modelled as Markov additive processes with boundaries. In Section 7.7, we propose an algorithm based on the adapted CFTP to sample perfectly from the stationary distribution of the delay process in an intree network.

7.2 Stochastic Recursive Sequence

We will introduce a *stochastic recursive sequence* which will be used for coupling from the past in the context of Monte Carlo simulation. We will assume that a sequence $\{(U_n^1, U_n^2)\}_{n=1}^{\infty}$ of bivariate independent uniforms can be generated. Customarily this is done with a pseudo-random number generator (PRNG). Knuth [K97] is a good reference for PRNGs. We will assume that the user as access to a good PRNG. See L'Ecuyer [L88] for an example of a good PRNG.

Fix T some positive integer. We want to start our Markov Chain in the past such that by time $n = 0$, the Markov chain will be in steady state. In other words, we want to generate $(X_{-T}, Y_{-T}), (X_{-T+1}, Y_{-T+1}), \ldots, (X_0, Y_0)$, such that (X_0, Y_0) is governed by the stationary distribution π.

Since we know the marginal distribution π_X, we can construct the transition kernel

$$K_X^*(x, x') := \frac{\pi_X(x')}{\pi_X(x)} K_X(x', x),$$

for the time reversed Markov chain X^*. We will generate $(X_{-T}, \ldots, X_{-1})$, i.e. the past trajectory, using the time reversed chain. Let ϕ_X^* be a measurable function that satisfies

$$P\left[\phi_X^*(x, U_n^1) = x'\right] = K_X^*(x; x').$$

The map ϕ_X^* is called an *update* rule. Note, an update rule is sometimes called a transition rule, see [F98]. An update rule is usually constructed using the well known "Inverse Transformation Technique", see [LK00, BFS87].

For any positive integer T and initial state $x_0 \in \mathcal{S}_X$, we may construct the time reversed Markov chain $\{X_n^*\}_{n=0}^T$, where $X_0^* = x_0$, with transition kernel K^* such that

$$X_0^* = x_0, \quad X_{n+1}^* = \phi_X^*(X_n^*, U_{n+1}^1), \text{ for all } n \in \{0, \ldots, T-1\}. \tag{7.2.1}$$

Let $X_n = X_{-n}^*$, for $n = -T, \ldots, 0$, i.e. $(X_{-T}, \ldots, X_0) = (X_0^*, \ldots, X_T^*)$. Note, that in steady-state, the one-step forward transitions of the trajectory $X_{-T}, \ldots, X_0$ are governed by the transition kernel K.

We use the latter trajectory to generate the remainder of the past. Let ϕ_Y be a measurable function that satisfies

$$P\left[\phi_Y(x, x', y, U_n^2) = y'\right] = K_Y(x, x', y; y').$$

For any initial $y \in \mathcal{S}_Y$, we may construct the stochastic sequence $\{Y_n(y)\}_{n=-T}^0$ with the update rule ϕ_Y as follows

$$Y_{-T} = y, \quad Y_{n+1} = \phi_Y(X_n, X_{n+1}, Y_n, U_{-n}^2), \text{ for all } n \in \{-T, \ldots, -1\}. \tag{7.2.2}$$

7.3 Coupling from the Past

In this section, we present a coupling from the past algorithm adapted to the stochastic recursive sequence presented in Section 7.2.

We introduce some notation. Let $x^* = \{x_0, x_{-1}, \ldots\}$ be some sample path for the reversed time Markov chain X^*. Let T be some positive integer, we define the vector $\vec{x}(T)$ associated to x^* as follows $\vec{x}(T) := (x_{-T}, \ldots, x_{-1}, x_0)$. The vector $\vec{x}(T)$ will be interpreted as a realisation for the random vector $(X_{-T}, \ldots, X_0)$.

We start by initializing $X_0 = x_0$, where $x_0 \in \mathcal{S}_X$ and initializing T as some positive integer. We generate the backward transitions of X using the update rule ϕ_X^* and the recurrence (7.2.1). This gives us some sample path $\vec{x}(T) = (x_{-T}, \ldots, x_{-1}, x_0)$.

Algorithm 1 Time Reversal and CFTP

Initialize $X_0 = x_0$, where $x_0 \in \mathcal{S}_X$.
Initialize T, a positive integer.
Initialize the seed to generate two independent sequences of pseudo-random numbers $U_1^1, U_2^1, \ldots$ and $U_1^2, U_2^2, \ldots$
for $n = 0$ to $T - 1$ **do**
  $x_{-n-1} \leftarrow \phi_X^*(x_{-n}, U_{n+1}^1)$
end for
repeat
  $i \leftarrow 0$
  for all $y \in \mathcal{S}_Y$ **do**
    $y[i] \leftarrow y$
    $i \leftarrow i + 1$
  end for
  for $n = -T$ to -1 **do**
    for $i = 0$ to $|\mathcal{S}_Y| - 1$ **do**
      $y[i] \leftarrow \phi_Y(x_n, x_{n+1}, y[i], U_{-n}^2)$
    end for
  end for
  $T \leftarrow T + 1$
  $x_{-T} \leftarrow \phi_X^*(x_{-T+1}, U_T^1)$
until $y[0] = y[1] = \cdots = y[|\mathcal{S}_Y| - 1]$
$Y_0 \leftarrow y[0]$
Output Y_0 is a realisation from the conditional distribution $\pi(x_0, \cdot)/\pi_X(x_0)$.
Note If X_0 is a random sample from π_X, then $(X_0, Y_0) \sim \pi$.

Similar to Propp and Wilson's general algorithm, we will run $|\mathcal{S}_Y|$ processes, one for each state in $\mathcal{S}_Y$. Let $y \in \mathcal{S}_Y$, we generate the process $\{Y_n\}$ as follows,

$$Y_{-T} = y, \quad Y_{n+1} = \phi_Y(x_n, x_{n+1}, Y_n, U_{-n}^2), \text{ for all } n \in \{-T, \ldots, -1\}. \tag{7.3.1}$$

Let $f_m^T(y)$ be the value of the process Y at time m, assuming we started at time $-T$ in the past and initialized the process as follows $Y_{-T} = y$. In other words, $f_m^T(y) = Y_m$, for $m = -T, \ldots, 0$. In particular, $f_0^T(y)$ represents the value of the process at time 0.

If $f_0^T(y)$ is constant for all $y \in \mathcal{S}_Y$ (i.e. all states have coalesced), then we stop and return the constant value, else we repeat the same procedure further in the past.

It should be noted that we do not need to regenerate $\vec{x}(T)$. We start with $\vec{x}(T) = (x_{-T}, \ldots, x_0)$ and go backwards, e.g.

$$\vec{x}(T+1) = (x_{-(T+1)}, x_{-T}, \ldots, x_0), \text{ where } x_{-(T+1)} = \phi_X^* \left(x_{-T}, U_{T+1}^1\right).$$

In other words, we always use the same x-value for a particular fixed time.

The latter coupling from the past adaptation is summarized in Algorithm 1.

The following theorems justify the correctness of Algorithm 1, i.e it will output a value Y_0 with probability 1 and that this value is a realisation of a random variable governed by the distribution $\pi(x_0, \cdot)/\pi_X(x_0)$. Moreover, if X_0 is a sample taken from the distribution π_X, then (X_0, Y_0) is a random variable governed by π.

Let $x_0 \in \mathcal{S}_X$. We say that $T \leq \infty$ is an x_0-restricted backward coupling time, if $f_0^T(y)$ is constant for all $y \in \mathcal{S}_Y$. In other words, starting at time $-T$ all of the $|\mathcal{S}_Y|$ processes will couple by time 0. The x_0-restricted backward coupling time T is called successful, if $T < \infty$.

Theorem 7.3.1 (Unique Output) *Let $x_0 \in \mathcal{S}_X$. Assume the existence of a successful x_0-restricted backward coupling time. Let T be the smallest of the x_0-restricted backward coupling times. All positive integers T' larger than T are also x_0-restricted backward coupling times and the output of Algorithm 1 is unique. In other words, $f_0^{T'}(y)$is constant, for all $y \in \mathcal{S}_Y$ and for all $T' \geq T$.*

Proof: Since T is an x_0-restricted backward coupling time, then $f_0^T(y)$ is constant for all $y \in \mathcal{S}_Y$. Let T' be some arbitrary integer larger than T. Regardless of the

value $f_T^{T'}(y)$ which is the value (at time T) of the process $\{Y_n\}$ that started at time $-T'$, the value of the process at time 0 will be constant. This follows from the fact that T is a x_0-restricted backward coupling time. From the arbitrariness of T', we obtain $f_0^{T'}(y)$ is constant, for all $y \in \mathcal{S}_Y$ and for all $T' \geq T$. This completes the proof.

□

In Section 7.5, we prove the following theorems using a censored (watched) Markov chain argument.

Theorem 7.3.2 *Let $x_0 \in \mathcal{S}_X$. With probability 1 there exists a successful x_0-restricted backward coupling time.*

Theorem 7.3.3 *Let $x_0 \in \mathcal{S}_X$. Assume that T is a successful x_0-restricted backward coupling time. Then $Y_0 = f_0^T(y)$ is a random variable with distribution $\mu(\cdot|x_0) := \pi(x_0, \cdot)/\pi_X(x_0)$, for arbitrary $y \in \mathcal{S}_Y$.*

Moreover, if x_0 is a realisation from the distribution π_X, then (X_0, Y_0) is governed by the distribution π.

7.4 Monotone Markov Chain

In this section, we adapt Algorithm 1 to the case where some *monotonicity* properties are satisfied.

We will assume that there exists a partial ordering on the state space $\mathcal{S}_Y$. This means, that there exists a relation $\preceq$ defined on $\mathcal{S}_Y$ that is a pre-order, i.e. reflexive ($y \preceq y$ for all $y \in \mathcal{S}_Y$) and transitive ($y \preceq y'$ and $y' \preceq y'' \Rightarrow y \preceq y''$), and that it is also anti-symmetric ($y \preceq y'$ and $y' \preceq y \Rightarrow y = y$). We denote the minimal and maximal states as $y_{\min}$ and $y_{\max}$, respectively. This partial ordering can also be defined on the total state space $\mathcal{S}$ as follows, if $y \preceq y'$, then $(x, y) \preceq (x, y')$.

We will require that the update rule ϕ_Y be $\preceq$-order preserving in the following sense,

$$\text{if } y \preceq y', \text{ then } \phi_Y(x, x', y, U_n^2) \preceq \phi_Y(x, x', y', U_n^2).$$

Algorithm 2 (Monotone - Time Reversal and CFTP):

Initialize $X_0 = x_0$, where $x_0 \in \mathcal{S}_X$.
Initialize T, a positive integer.
Initialize the seed to generate two independant sequences of pseudo-random numbers $U_1^1, U_2^1, \ldots$ and $U_1^2, U_2^2, \ldots$
for $n = 0$ to $T - 1$ **do**
  $x_{-n-1} \leftarrow \phi_X^*(x_{-n}, U_{n+1}^1)$
end for
repeat
  Initialize $y[0] = y_{\min}$ and $y[1] = y_{\max}$
  for $n = -T$ to -1 **do**
    $y[0] \leftarrow \phi_Y(x_n, x_{n+1}, y_{\min}, U_{-n}^2)$
    $y[1] \leftarrow \phi_Y(x_n, x_{n+1}, y_{\max}, U_{-n}^2)$
  end for
  if $y[0] \neq y[1]$ **then**
    for $n = T$ to $2T - 1$ **do**
      $x_{-n-1} \leftarrow \phi_X^*(x_{-n}, U_{n+1}^1)$
    end for
    $T \leftarrow -2T$
  end if
until $y[0] = y[1]$
$Y_0 \leftarrow y[0]$
Output Y_0 is a realisation from the conditional distribution $\pi(x_0, \cdot)/\pi_X(x_0)$.
Note If X_0 is a random sample from π_X, then $(X_0, Y_0) \sim \pi$.

Under these monotonicity properties, we only have to run two processes. One process will be initialized at the minimum state and the other at the maximum state. Since the update rule preserves the order of the partial order, the latter processes form bounding processes. The current states of all the other processes are always going to fall between the current states of the two bounding processes. Hence, once the bounding processes have coupled the states of all processes will have coalesced.

Algorithm 2 is an adaptation of the algorithm proposed by Propp and Wilson [PW96]. Notice that the step backward is always doubled. This doubling was proposed by Propp and Wilson [PW96] for efficiency. It is a consequence of running two processes.

7.5 Censored Markov Chain

In this section we construct a censored (watched) Markov chain $\tilde{Y}$ defined by all return times of the process X to the state x_0. The censored Markov chain will allow us to reduce our problem to the setting of Propp and Wilson [PW96], that is an ergodic Markov chain with finite state space. We will then be able to apply their results to prove Theorems 7.3.2 and 7.3.3.

We state a well known result for censored Markov chains, see [KSK76, GH90].

Lemma 7.5.1 *Let $\{W_n\}$ be a Markov chain with transition kernel K_W and state space $\mathcal{S}_W$. Consider the subset of states $E \subset \mathcal{S}_W$. If $\{W_n\}$ is an irreducible Markov chain, then the Markov chain $\{\tilde{W}_n\}$ generated by the transition kernel*

$$\tilde{K}_W = T + L\,\hat{Q}\,U, \tag{7.5.1}$$

where

$$K_W = \begin{array}{c} \\ E \\ E^c \end{array} \begin{array}{c} \begin{array}{cc} E & E^c \end{array} \\ \begin{bmatrix} T & U \\ L & Q \end{bmatrix} \end{array} \quad \text{and} \quad \hat{Q} = \sum_{k=0}^{\infty} Q^k,$$

is also an irreducible Markov chain. Moreover, if $\{W_n\}$ is an ergodic Markov chain with stationary distribution π_W, then $\{\tilde{W}_n\}$ is also ergodic with stationary distribution

$$\tilde{\pi}(w) = \frac{\pi_W(w)}{\pi_W(E)}, \qquad \text{for } w \in E.$$

Consider the renewal sequence $\{T_m^{x_0}\}$ where

$$T_0^{x_0} = 0 \quad \text{and} \quad T_m^{x_0} = \min\{n > T_{m-1}^{x_0} : X_n^* = x_0\},$$

that is the sequence of return times to x_0. Since X^* is positive recurrent then the mean increment for the renewal sequence is finite, $E[T_m^{x_0} - T_{m-1}^{x_0}] < \infty$. The associated counting renewal process is

$$N^{x_0}(n) = \max\{m : T_m \leq n\},$$

the number of return times to x_0 up to time n.

Let T be some positive integer. Consider the hitting time

$$\tau_{x_0}(T) = \min\{m : T_m^{x_0} > T\},$$

the smallest renewal time greater than T. By Theorem 7.3.1, if T is an x_0-restricted backward coupling time then the hitting time $\tau_{x_0}(T)$ is also an x_0-restricted backward coupling time.

Let $y \in \mathcal{S}_Y$. We construct the censored Markov chain $\tilde{Y}$ as follows. Consider the process Y that we start at time $-\tau_{x_0}(T)$ in the past. The process $\tilde{Y}$ will be the value of Y as we watch the process (X, Y) on $\{x_0\} \times \mathcal{S}_Y$, that is

$$\tilde{Y}_{-m} = f_{-T_m}^{\tau_{x_0}(T)}(y) = Y_{-T_m}, \qquad \text{for } m = -N^{x_0}(\tau_{x_0}(T)), \ldots, 0.$$

The process $\tilde{Y}$ is a Markov chain with finite state space and transition kernel $\tilde{K}$. We will show that $\tilde{K}$ is of the form (7.5.1) with $K_W = K$. Consequently, by Lemma 7.5.1 the Markov chain $\tilde{Y}$ has stationary distribution $\mu(\cdot|x_0) = \pi(x_0, \cdot)/\pi_X(x_0)$.

Let $\{(x_0, y), (x_{-N+1}, y_{-N+1}), \ldots, (x_{-1}, y_{-1}), (x_0, y')\}$ be one possible sample path from the past for the Markov chain (X, Y) that begins at (x_0, y), ends at (x_0, y') and such that $x_n \neq x_0$ for all $n = -N+1, \ldots, -1$. By construction, the probability of this sample path is

$$K_X^*(x_0, x_{-1}) \cdots K_X^*(x_{-T+1}, x_0) \times$$
$$K_Y(x_0, x_{-T+1}, y; y_{-T+1}) \cdots K_Y(x_{-1}, x_0, y_{-1}; y').$$

Since

$$K^*(x;x') = \frac{\pi_X(x')}{\pi_X(x)} K_X(x';x)$$

and

$$K(x,y;x',y') = K_X(x;x')\, K_Y(x,x',y;y'),$$

then after simplification the probability of the sample path becomes

$$K(x_0,y;x_{-N+1},y_{-N+1}) \cdots K(x_{-1},y_{-1};x_0,y').$$

Sum over all possible sample values $\{(x_{-T+1},y_{-T+1}),\cdots,(x_{-1},y_{-1})\}$ such that $x_n \neq x_0$ for $-T+1 \leq n \leq -1$ to obtain

$$\tilde{K}(y;y') = (T + L\,\hat{Q}\,U)(y;y'),$$

where

$$K = \begin{array}{c} \\ E \\ E^c \end{array} \begin{array}{c} \begin{array}{cc} E & E^c \end{array} \\ \begin{bmatrix} T & U \\ L & Q \end{bmatrix} \end{array} \quad \text{and} \quad \hat{Q} = \sum_{k=0}^{\infty} Q^k,$$

and $E = \{x_0\} \times \mathcal{S}_Y$. It follows from Lemma 7.5.1 that $\tilde{Y}$ has stationary distribution $\mu(\cdot|x_0) := \pi(x_0,\cdot)/\pi_X(x_0)$.

We now state two lemmas that follows directly from the results in [PW96] and prove Theorems 7.3.2 and 7.3.3.

Lemma 7.5.2 *Let $x_0 \in \mathcal{S}_X$. With probability 1 there exists a hitting time τ_{x_0} such that $\tilde{Y}_0 = f_0^{\tau_{x_0}}(y)$ is constant for all $y \in \mathcal{S}_Y$.*

Lemma 7.5.3 *Consider the hitting time τ_{x_0}, such that $\tilde{Y}_0 = f_0^{\tau_{x_0}}(y)$ is constant for all $y \in \mathcal{S}_Y$, then the random variable $\tilde{Y}_0$ is governed by the distribution $\mu(\cdot|x_0) := \pi(x_0,\cdot)/\pi_X(x_0)$.*

Proof of Theorem 7.3.2: Let $T = \tau_{x_0}$ from Lemma 7.5.2, then T is a successful x_0-restricted backward coupling time. Such a T exists almost surely. This completes the proof.

□

Proof of Theorem 7.3.3: Let T be an x_0-restricted backward coupling time. Consider the sequence of return times to x_0 greater than T, i.e. $\{T_m^{x_0}\}_{m=m'}^{\infty}$, where $m' = \min\{m : T_m^{x_0} > T\}$.

Consider a realisation of the independent uniform random variables ω. From the Unique Output Theorem 7.3.1, $f_0^{T_m^{x_0}}(y)(\omega)$ is constant for all $m \geq m'$. Let $\tilde{Y}_{-\infty}(\omega)$ be this constant. Thus,

$$\lim_{m\to\infty} f_0^{T_m^{x_0}}(y)(\omega) = \tilde{Y}_{-\infty}(\omega).$$

From Lemma 7.5.3, each of the random variables $\{f_0^{T_m^{x_0}}(y) : m \geq m'\}$ is governed by $\mu(\cdot|x_0) = \pi(x_0, \cdot)/\pi_X(x_0)$, thus $\tilde{Y}_{-\infty}$ is governed by the distribution $\mu(\cdot|x_0)$.

Since T is an x_0-restricted backward coupling time, then by the Unique Output Theorem 7.3.1, $f_0^T(y) = Y_{-\infty}$, for all $y \in \mathcal{S}_Y$. Therefore, for all $y \in \mathcal{S}_Y$, $f_0^T(y)$ is governed by $\mu(\cdot|x_0)$. This completes the proof.

□

7.6 Adapted CFTP and Queueing Processes

In the following subsections, we will present some examples of processes to which our adapted CFTP algorithm can be applied.

Consider an aperiodic and irreducible Markov chain M with a countable or finite state space. Consider an integer valued process Z such that (Z, M) is a Markov chain with transition kernel of the form

$$P_{m,m'}(z) = \mathsf{P}\left[Z_1 \leq a, M_1 = m' \mid M_0 = m\right].$$

We model the queue increments, i.e. arrivals minus departures, as as the process Z.

The number of customers buffered at the end of the time slot n is denoted Q_n. We will assume a finite capacity buffer of L customers and that the process Q is governed by the following recurrence

$$Q_n = \max\left(\min\left(Q_{n-1} + Z_n, L\right), 0\right).$$

As an example of such a queue, we could consider an ATM-multiplexer with independent *on-off* sources with geometric sojourn times for the alternating *on* and

off periods. If the sojourn times were heavy tailed rather than geometric, as in [R02], then we would have an example with a denumerable state space $\mathcal{S}_X$, where $X = M$.

To be able to apply the adapted monotone - CFTP Algorithm 2, we must show that there exists a partial ordering $(\preceq, \mathcal{S}_Y)$ and an update rule ϕ_Y that is $\preceq$-order preserving, i.e.

$$\text{if } y \preceq y' \quad \text{then} \quad \phi_Y(x, x', y, U_n^2) \preceq \phi_Y(x, x', y', U_n^2).$$

7.6.1 Single Class Queue

Consider a single class queue, see Figure 5. The process $Y = Q$ will represent the queue size. Since the buffer has a finite capacity L, then the state space $\mathcal{S}_Y$ is finite. The minimal and maximal states are $y_{\min} = 0$ and $y_{\max} = L$, respectively.

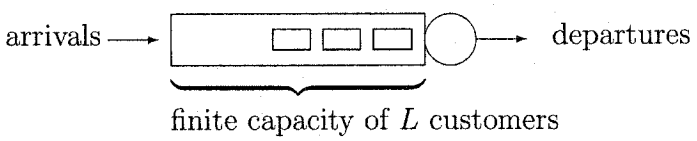

Figure 5: Single Class Queue

We utilize the queue size to define the partial ordering on $\mathcal{S}_Y$. Define $(\preceq, \mathcal{S}_Y)$ as follows :

$$q \leq q' \implies y \preceq y'. \tag{7.6.1}$$

From a given state (x_0, q_0) and one-step transition in X, (x_0, x_1), we utilize the update rule ϕ_Y to obtain the next queue size. Let $q_1 = \phi_Y(x_0, x_1, q_0, U_n^2)$, i.e. the queue size after one transition. The queue sizes must satisfy

$$q_1 = \max\left(\min\left(q_0 + z(x_0, x_1), L\right), 0\right),$$

where $z(x_0, x_1)$ represents the increment. Now choose another queue size q_0' larger than or equal to q_0. Let $q_1' = \phi_Y(x_0, x_1, q_0', U_n^2)$. Thus,

$$q_1' = \max\left(\min\left(q_0' + z(x_0, x_1), L\right), 0\right).$$

It should be clear that if $q_0 \leq q_0'$ then $q_1 \leq q_1'$, thus the update rule ϕ_Y is $\preceq$-order preserving. Therefore, the single class process gives a monotone Monte Carlo algorithm for approximating π, to use the terminology from [PW96]. In other words, we can apply Algorithm 2.

7.6.2 Multiclass Queue

Consider a multiclass class queue, see Figure 6. The process Y is not just the queue size in this case, that is $Y = (Q, C)$ where Q represents the queue size and C represents the class of the customers in the buffer. The classes are used for routing.

The process C is a sequence of random vectors $C_n = (C_n^1, \ldots, C_n^L)$, where C_n^i represents the class of the ith customer in the buffer at the end of the nth time slot. We use the class 0 to denote an empty slot. Thus, $C_n = \underline{0}$ means $Q_n = 0$.

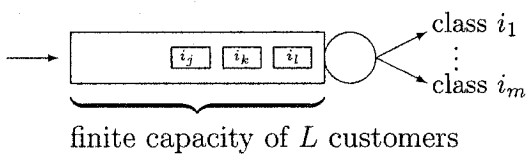

Figure 6: Multiclass Queue

We will utilize the same ordering as in the single class case. Define the ordering $(\preceq, \mathcal{S}_Y)$ as follows. For any two states $y_0 = (q_0, \underline{c}_0)$ and $y_0' = (q_0', \underline{c}_0')$,

$$q_0 \leq q_0' \Longrightarrow y \preceq y'.$$

Remark : The ordering $(\preceq, \mathcal{S}_Y)$ is not a partial order, but it is a pre-order. It is not anti-symmetric, since we can have $q_0 = q_0'$ without $\underline{c}_0 = \underline{c}_0'$.

Using the same arguments as in the single class case, we can show that the update rule ϕ_Y is $\preceq$-order preserving. Although, we will not be able to apply Algorithm 2 directly in this case. We do not have a unique maximum state $y_{\max}$. This means that, if we run a maximum process and a minimum process and they couple, then not all

the processes have necessarily coupled. To get around this problem we use the fact that we have a unique minimum $y_{\min} = (0, \underline{0})$.

We want to be able to identify the coupling of all the processes without having to simulate them all. We will argue now that if we initialize a process at any of the maximal states $y_{\max}$, where

$$y_{\max} \in \{y \in \mathcal{S}_y : y' \preceq y, \text{ for all } y' \in \mathcal{S}_Y\},$$

then, when this process hits $y_{\min} = (0, \underline{0})$, all of the processes will have coupled.

Consider two processes Y^1 and Y^2. At time $-T$, the process Y^1 is initialized to one of the maximal states, denoted by y_{max}. Let $Y^2_{-T} = y$, where y is some arbitrary state in $\mathcal{S}_Y$. Since $y_{\max}$ is a maximal state, then $y \preceq y_{\max}$. Note, we could also have $y_{\max} \preceq y$, since we do not have a unique maximal state. Now, suppose that at time $-t$, the process Y^1 hits the minimal state, i.e. $Y^1_{-t} = y_{\min} = (0, \underline{0})$. The update rule is $\preceq$-preserving, thus if $Y^2_{-t} = y'$, then we must have $y' \preceq y_{\min}$. But $y_{\min}$ is the minimal state, thus y' is also the minimal state, that is $y' = y_{\min} = (0, \underline{0})$. Hence, the processes have coupled. From the arbitrariness of y, we can conclude that all processes have coupled by time $-t$, i.e. when the process Y^1 hits $(0, \underline{0})$.

The new algorithm is presented in Algorithm 3. The inspiration for this algorithm comes from [TC00]. In the latter reference, the authors present a CFTP algorithm for a queueing model in a general state space setting. Their key idea is to wait for a time at which all paths are in some "small set". Similarly, we are waiting for a time at which all the processes are at the minimal state $y_{\min} = (0, \underline{0})$.

7.6.3 Acyclic Feedforward Intree Network

Consider an acyclic feedforward intree network, see [CHJS94]. An example of such a network is given by Figure 7. The output stream from a node, in such a network, does not split. Thus, we have a single class network. The nodes are partitioned by level. The nodes at level n feed the buffers at level $n + 1$. Note, all departures from the network occur at the most downstream node. We assume that the service times do not depend on the head-of-the-line.

Here the process $Y = (\vec{Q})$ represents the vector of the queue sizes. As a first guess, we could try a partial ordering, $(\preceq, \mathcal{S}_Y)$, based on the total number of customers in the network. This construction is similar to the partial ordering for the single class queue. The problem with this construction is that the update rule ϕ_Y would not be $\preceq$-order preserving.

Consider the following example. We are running two processes Y^1 and Y^2. Assume that at the end of some time slot t, that Y_t^1 equals some state that has a full buffer at a node on level 1. Suppose that during the next transition, there are multiple arrivals at this node. Also assume that Y_t^2 equals some state that will accommodate all those arrivals. The change in total number of customers in the network could be larger for Y^2. If the Y^1 had only one extra customer than Y^2 before the transition, then after the transition Y^2 could have more customers than Y^1. This means that the update rule would not be $\preceq$-order preserving.

To work around this problem we can consider the queue sizes individually. Let $|B|$ the total number of nodes. We assume that the states in $\mathcal{S}_Y$ are of the form $y = (q_1, q_2, \ldots, q_{|B|})$, where q_b is the queue size at node b.

We define the partial ordering $(\preceq, \mathcal{S}_Y)$ as follows :

$$q_b \leq q'_b, \text{ for all } b \in \{1, \ldots, |B|\} \quad \Longrightarrow \quad y \preceq y'.$$

We now argue that the update rule ϕ_Y is $\preceq$-order preserving.

For the single class queue we showed that for a partial ordering based on the queue size the update rule is order preserving. In other words, if we initialize a queue at its maximum, then it will remain stochastically larger than the other queues since they all have the same arrivals.

In the present intree network not all the queues will have the same arrivals, since some arrivals will come from the upstream queues. Although, it should be remarked that the service slots occur at the same time for all the processes.

We now proceed with an inductive argument. Consider two processes Y^1 and Y^2. We will assume that initially the queue sizes at all nodes are larger for Y^1. The queue sizes at the nodes from level 1 are stochastically larger for Y^1 compared to the same nodes for Y^2. This is our base case. Our induction hypothesis is that the queue sizes

at the nodes from level n are stochastically larger for Y^1 compared to the same nodes for Y^2. Consider the queues from level $n+1$. The arrivals depend on the external sources and the upstream buffers from level n. For both processes the number of arrivals from the external sources are the same, but the number of arrivals from the upstream buffers can be different if one of the buffers is empty, or almost empty. We assumed that the queues from level n were larger for Y^1, thus the arrival process for each node from level $n+1$ will be stochastically larger for Y^1. Thus, the queue sizes for the nodes from level $n+1$ must be stochastically larger for Y^1.

The latter inductive argument demonstrates that we can use Algorithm 2 for the acyclical feedforward intree network.

7.7 Delay Process

Consider the intree network from Section 7.6.3. Suppose that we are interested in the end-to-end delays. We must extend the state space $\mathcal{S}_Y$ to include the age of each customer in the network.

The states in $\mathcal{S}_Y$ are of the form $y = (\underline{q}, \underline{d})$, where $d_b(i)$ represents the delay for ith customer waiting at node b and q_b represents the queue size at node b. If there are less than i customers waiting at node b, then we will let $d_b(i) = 0$. Thus, the empty network is represented by $(\underline{0}, \underline{0})$.

We show that this delay process gives a monotone Monte Carlo algorithm for approximating π. The arguments utilized in this section will be similar to those used in the multiclass queue setting of Section 7.6.2.

We define the pre ordering $(\preceq, \mathcal{S}_Y)$ as follows :

$$q_b \leq q'_b, \text{ for all } b \in \{1, \ldots, |B|\} \quad \Longrightarrow \quad y \preceq y'.$$

This pre ordering induces a unique minimum state $y_{\min} = (\underline{0}, \underline{0})$ which is an empty network, i.e. no customers in the network. But we do not have a unique maximum since the ordering only depends on the queue sizes and not the ages.

We demonstrated in Section 7.6.3 that the update rule ϕ_Y is $\preceq$-order preserving. Similar to the multiclass queue, we can initialize the delay process, at time $-T$, to

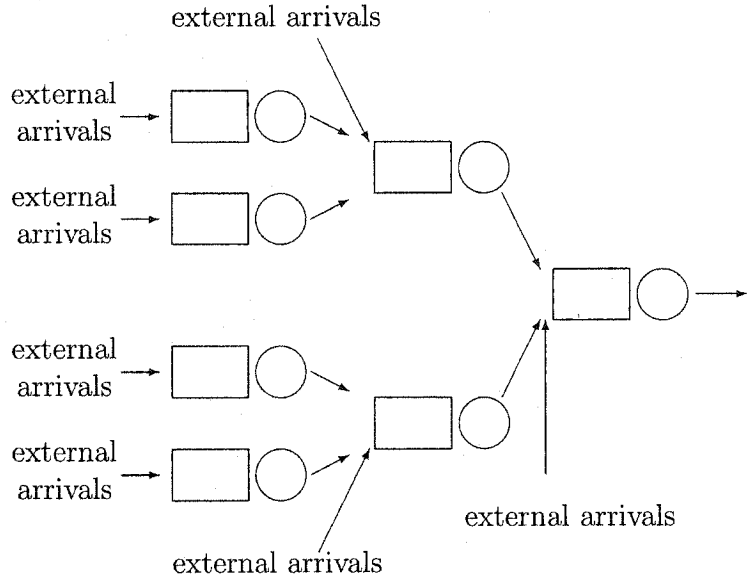

Figure 7: Acyclical Feedforward Intree Network

one of its maximal states $y_{\max}$. If this process hits the unique minimal state $(\underline{0}, \underline{0})$, then all the other processes must also be at the minimal state $(\underline{0}, \underline{0})$. Thus, we can use the adapted Algorithm 3 to sample from the stationary distribution π of the delay process.

Remark: When initializing the process Y in the past, at time $-T$, we should only be concerned with the queue sizes, i.e. full buffers, and not worry about the delays until the network empties. The reason is that the pre ordering $\preceq$ ignores the delays. Although, once the process hits $(\underline{0}, \underline{0})$, that is the network empties, it is important to store the customer arrival times, since we want to sample from the stationary distribution of the delay process.

Algorithm 3 (Pre-order with a Unique Minimum - Time Reversal and CFTP):

Initialize STOP $\leftarrow$ FALSE
Initialize $X_0 = x_0$, where $x_0 \in \mathcal{S}_X$.
Initialize T, a positive integer.
Initialize the seed to generate two independant sequences of pseudo-random numbers $U_1^1, U_2^1, \ldots$ and $U_1^2, U_2^2, \ldots$
for $n = 0$ to $T - 1$ **do**
  $x_{-n-1} \leftarrow \phi_X^*(x_{-n}, U_{n+1}^1)$
end for
repeat
  Initialize $y = y_{\max}$, where $y_{\max}$ is one of the maximal states
  for $n = -T$ to -1 **do**
    $y \leftarrow \phi_Y(x_n, x_{n+1}, y, U_{-n}^2)$
    if $y = y_{\min}$ **then** STOP $\leftarrow$ TRUE
  end for
  if STOP = FALSE **then**
    for $n = T$ to $2T - 1$ **do**
      $x_{-n-1} \leftarrow \phi_X^*(x_{-n}, U_{n+1}^1)$
    end for
    $T \leftarrow -2T$
  end if
until STOP = TRUE
$Y_0 \leftarrow y$
Output Y_0 is a realisation from the conditional distribution $\pi(x_0, \cdot)/\pi_X(x_0)$.
Note If X_0 is a random sample from π_X, then $(X_0, Y_0) \sim \pi$.

Chapter 8

Concluding Remarks

We found a workload process correlated to the end-to-end delay. We called it the encumbrance. The latter can be modelled as a Markov additive chain with boundary. The encumbrance is exponentially twisted using a well known h-transform technique. We showed that the h-transform does exist under mild conditions. We also obtained a large deviation result for the encumbrance process in a two node intree tandom network with infinite buffer capacities. We do not have a large deviation result for a more general ATM-type network, nonetheless the change of measure still exists.

This change of measure was used to propose the importance sampling estimator within the context of the A-cycle simulation method. We also showed that the associated likelihood ratio function is easily computed. It only depends on the state of the process at the beginning of an A-cycle and at the time when we untwist the process and it also depends on the frequencies of returns to the edge Δ and the frequencies of cell losses due to buffer overload during the twisted phase of the A-cycle.

We have constructed an importance sampling estimator to estimate the probability of extreme cell delays for a particular virtual connection through an ATM-type network. We showed empirically with a simulation that this estimator can be much more effective than a crude Monte-Carlo estimator. In this particular example we were able to estimate a probability in the order of 10^{-8} with a coefficient of variation of about 15%. To estimate the probability of such a rare event with a crude Monte-carlo estimator would be practically impossible. The computational effort required

to reduce the coefficient of variation in a reasonable range is prohibitively large in the crude Monte-Carlo case. The simulated network in our example was a two node intree tandem network. It would be interesting to see if our importance sampling estimator is still efficient for much larger ATM-type networks.

It remains to see if conditions could be found for asymptotic efficiency or optimality of the importance sampling estimator or maybe for a closely related estimator. Another interesting question concerns the internal switching of the cells within the network. The ATM-type network as defined in this thesis does not allow the feedback of cells to previous levels in the network. This means that a cell could pass through a tagged node more than once. Could the definition of the encumbrance be modified to allow for such a scenario and still preserve the Markov additive structure?

We ended the thesis with a perfect sampling result. When performing ratio estimation with the A-cycle method, we must have a burn in phase to lessen the initial bias. We proposed a variant of the coupling from the past algorithm of Propp and Wilson to sample exactly from the stationary distribution of the process. By weakening the Propp and Wilson assumption of partial ordering to an assumption of pre ordering, we were able to adapt their algorithm to sample exactly from the stationary distribution of the delay process in the intree network case. An open problem remains. Can we sample exactly from the stationary distribution when the network is non intree?

Appendix A

Proofs and Technical Results

Proof of Lemma 4.5.3: The function $V_b^\gamma(w) = \exp\{\theta\, e - \gamma\, q_b\}\, \hat{a}_\gamma^{*b}(\hat{w})$ is a positive eigenvector for K^∞ with eigenvalue $\exp(\Lambda_b(s))$. In other words,

$$\sum_{w' \in S^\infty} K^\infty(w; w')\, \exp\{\theta\, e' - \gamma\, q_b'\}\, \hat{a}_\gamma^{*b}(\hat{w}') = \exp(\Lambda_b(s))\, \exp\{\theta\, e - \gamma\, q_b\}\, \hat{a}_\gamma^{*b}(\hat{w}).$$

Divide both sides by $e^{\theta\, e}\, \hat{a}(\hat{w})$. We obtain

$$\sum_{w' \in S^\infty} K^\infty(w; w')\, \frac{e^{\theta\, e'}\, \hat{a}(\hat{w}')}{e^{\theta\, e}\, \hat{a}(\hat{w})}\, \exp\{-\gamma\, q_b'\}\, \frac{\hat{a}_\gamma^{*b}(\hat{w}')}{\hat{a}(\hat{w}')} \;=\; \exp(\Lambda_b(s))\, \exp\{-\gamma\, q_b\}\, \frac{\hat{a}_\gamma^{*b}(\hat{w})}{\hat{a}(\hat{w})}.$$

Since $\mathcal{K}^\infty(w; w') = K^\infty(w; w')\, h(w')/h(w)$ and $h(w) = e^{\theta\, e}\, \hat{a}(\hat{w})$, we obtain

$$\sum_{w' \in S^\infty} \mathcal{K}^\infty(w; w')\, \exp\{-\gamma\, q_b'\}\, \frac{\hat{a}_\gamma^{*b}(\hat{w}')}{\hat{a}(\hat{w}')} \;=\; \exp(\Lambda_b(s))\, \exp\{-\gamma\, q_b\}\, \frac{\hat{a}_\gamma^{*b}(\hat{w})}{\hat{a}(\hat{w})}.$$

This is the desired result.

□

Proof of Lemma 4.5.2: We will use the notation as found in Section 4.4, i.e.

$$O_B(w) = \{b \in TN : q_b = 0, v_b = 0\} \quad \text{and} \quad \zeta_\Delta(w) = \sum_{b \in O_B(w)} d_b.$$

We will need the following indicator functions:

$$\vartheta_\Delta^b(w) = \begin{cases} 1, & \text{if } q_b = 0, v_b = 0 \\ 0, & \text{otherwise} \end{cases}$$

Note, $w \in \mathcal{G}_\Delta^b$ if and only if $\vartheta_\Delta^b(w) = 1$, since

$$\mathcal{G}_\Delta^b = \{w \in \Delta \ : \ q_b = 0, v_b = 0\}.$$

Also, if $w \in \mathcal{G}_\Delta^1$ and $q_2 > 0$, then $\zeta_\Delta(w) = d_1$, if $w \in \mathcal{G}_\Delta^2$ and $q_1 > 0$, then $\zeta_\Delta(w) = d_2$, and if $w \in S \setminus \mathcal{G}_\Delta$, then $\zeta_\Delta(w) = 0$.

From these observations, we can conclude that (4.5.6), (4.5.7), and (4.5.8) are equivalent to

$$\sum_{w'} K(w, w')\, e^{\theta\,(e'-e)-s\,(q_1'-q_1)} \hat{a}_s^{*1}(\hat{w}') = e^{\theta\,\zeta_\Delta(w)-s\,\vartheta_\Delta^1(w)+\Lambda_1(s)}\, \hat{a}_s^{*1}(\hat{w}), \tag{A.0.1}$$

and

$$\sum_{w'} K(w, w')\, e^{\theta\,(e'-e)-s\,(q_2'-q_2)} \hat{a}_s^{*2}(\hat{w}') = e^{\theta\,\zeta_\Delta(w)-s\,\vartheta_\Delta^2(w)+s\vartheta_\Delta^1+\Lambda_2(s)}\, \hat{a}_s^{*2}(\hat{w}), \tag{A.0.2}$$

since $V_b^s(w) = \exp\{\theta\, e - s\, q_b\}\, \hat{a}_s^{*b}(\hat{w})$, where $\hat{a}_s^{*1}$ and $\hat{a}_s^{*2}$ are defined by (4.5.23) and (4.5.30), respectively.

We will prove (A.0.1) and (A.0.2) in two steps: i) (A.0.1) that is $b = 1$ and ii) (A.0.2) that is $b = 2$.

Case i: The increments for the total encumbrance ENC_t and the queueing process for the first node Q_t^1 are

$$\Delta\, ENC_t = \sum_{a \in A_1 \cup A_2} \varepsilon_a\, A_t^a - \sum_{b \in TN} d_b\, B_t^b + \zeta_\Delta(W[t-1]), \tag{A.0.3}$$

and

$$\Delta\, Q_t^1 = \left(\sum_{a \in A_1} A_t^a \right) - B_t^1 + \vartheta_\Delta^1(W[t-1]). \tag{A.0.4}$$

Recall, A_b is the index set for the sources that directly offer cells to node b, A_t^a is the indicator function which is 1 when source a offers a cell during the time slot t, and B_t^b is the indicator function which is 1 when a cell is served at node b during the time slot t. We assume that

$$B_t^b = \mathbf{1}\left(V_{t-1}^b = 0\right),$$

i.e. a cell is served during a service slot even if there are no cells to serve. The indicator will compensate ϑ_Δ^1 for this service error.

The right-hand-side of (A.0.1) is equal to

$$
\begin{aligned}
& \sum_{w'} K(w,w')\, e^{\theta\,(e'-e)-s\,(q_1'-q_1)}\hat{a}_s^{*1}(\hat{w}') \\
= & \sum_{w'} E_w\left[e^{\theta\,\Delta ENC_1-s\,\Delta Q_1^1}\,\mathbf{1}\left(W[1]=w'\right)\right]\,\hat{a}_s^{*1}(\hat{w}') \\
= & \; e^{\theta\,\zeta_\Delta(w)-s\,\vartheta_\Delta^1(w)}\sum_{w'} E_w\left[\mathbf{1}\left(W[1]=w'\right)\times\right. \\
& \left.\exp\left\{(\varepsilon_a\,\theta-s)\sum_{a\in A_1}A_1^a+\varepsilon_a\,\theta\sum_{a\in A_2}A_1^a+(s-d_1\,\theta)\,B_1^1-d_2\,\theta\,B_1^2\right\}\right]\,\hat{a}_s^{*1}(\hat{w}').
\end{aligned}
$$

The last equation follows from (A.0.3) and (A.0.4). Since the processes A^a for $a \in A_1 \cup A_2$ and B^b for $b = 1, 2$ are conditionally independent and that they only depend on $\hat{w}$, summing over $\tilde{w}$ we obtain

$$
\begin{aligned}
& e^{\theta\,\zeta_\Delta(w)-s\,\vartheta_\Delta^1(w)}\times \\
& \prod_{a\in A_1}\sum_{m_a,u_a} E_{m_a,u_a}\left[\exp\left\{(\varepsilon_a\,\theta-s)\,A_1^a\right\}\,\mathbf{1}(M_1^a=m',U_1^a=u')\right]\,\hat{a}_s^{*1}(m_a',u_a')\times \\
& \prod_{a\in A_2}\sum_{m_a,u_a} E_{m_a,u_a}\left[\exp\left\{\varepsilon_a\,\theta\,A_1^a\right\}\,\mathbf{1}(M_1^a=m',U_1^a=u')\right]\,\hat{a}_\theta^a(m_a',u_a')\times \\
& E_{v_1}\left[\exp\left\{(s-d_1\,\theta)\,B_1^1\right\}\,\mathbf{1}(V_1^1=v_1')\right]\,\hat{a}_s^{*1}(v_1')\times \\
& E_{v_2}\left[\exp\left\{-d_2\,\theta\,B_1^2\right\}\,\mathbf{1}(V_1^2=v_2')\right]\,\hat{a}_{-\theta}^2(v_2') \\
= & \; e^{\theta\,\zeta_\Delta(w)-s\,\vartheta_\Delta^1(w)}\times\prod_{a\in A_1}e^{-\alpha_a^*(s)}\,\hat{a}_s^{*a}(m_a,u_a)\times\prod_{a\in A_2}e^{-\alpha_a(\theta)}\,\hat{a}_\theta^a(m_a,u_a)\times \\
& e^{-\beta_1^*(s)}\,\hat{a}_s^{*1}(v_1')\times e^{-\beta_2(-\theta)}\,\hat{a}_{-\theta}^2(v_2).
\end{aligned}
$$

The last equality follows from Lemma 3.5.1, Lemma 3.4.3, Lemma 3.6.2 and Lemma 3.6.1.

Since

$$
\Lambda_1(s)=\sum_{a\in A_1}-\alpha_a^*(s)+\sum_{a\in A_2}-\alpha_a(\theta)-\beta_1^*(s)-\beta_2(-\theta),
$$

then it follows that

$$
\sum_{w'} K(w,w')\, e^{\theta\,(e'-e)-s\,(q_1'-q_1)}\hat{a}_s^{*b}(\hat{w}') = e^{\theta\,\zeta_\Delta(w)-s\,\vartheta_\Delta^1(w)+\Lambda_1(s)}\,\hat{a}_s^{*1}(\hat{w}).
$$

This proves the case (i).

Case ii: The increments for the queueing process for the first node Q_t^2 are

$$\Delta Q_t^2 = \left(\sum_{a \in A_2} A_t^a \right) + B_t^1 - \vartheta_\Delta^1(W[t-1]) - B_t^2 + \vartheta_\Delta^2(W[t-1]). \tag{A.0.5}$$

Recall, A_2 is the index set for the sources that directly offer cells to node 2, A_t^a is the indicator function which is 1 when source a offers a cell during the time slot t, and B_t^b is the indicator function which is 1 when a cell is served at node b during the time slot t. We assume that

$$B_t^b = \mathbf{1}\left(V_{t-1}^b = 0\right),$$

i.e. a cell is served during a service slot even if there are no cells to serve. The indicator will compensate ϑ_Δ for this service error.

The right-hand-side of (A.0.2) is equal to

$$\begin{aligned}
&\sum_{w'} K(w,w')\, e^{\theta\,(e'-e)-s\,(q_2'-q_2)} \hat{a}_s^{*2}(\hat{w}') \\
= &\sum_{w'} E_w \left[e^{\theta\,\Delta ENC_1 - s\,\Delta Q_1^2}\, \mathbf{1}\left(W[1]=w'\right) \right] \hat{a}_s^{*2}(\hat{w}') \\
= &\; e^{\theta\,\zeta_\Delta(w) - s\,\vartheta_\Delta^2(w) + s\,\vartheta_\Delta^1(w)} \sum_{w'} E_w \left[\mathbf{1}\left(W[1]=w'\right) \exp\left\{ \varepsilon_a\,\theta \sum_{a \in A_1} A_1^a \right\} \times \right. \\
&\left. \exp\left\{ (\varepsilon_a\,\theta - s) \sum_{a \in A_2} A_1^a + (-s - d_1\,\theta)\, B_1^1 + (s - d_2\,\theta)\, B_1^2 \right\} \right] \hat{a}_s^{*2}(\hat{w}').
\end{aligned}$$

The last equation follows from (A.0.3) and (A.0.5). Since the processes A^a for $a \in A_1 \cup A_2$ and B^b for $b = 1, 2$ are conditionally independent and that they only depend on $\hat{w}$, summing over $\tilde{w}$ we obtain

$$
\begin{aligned}
& e^{\theta\,\zeta_\Delta(w) - s\,\vartheta^2_\Delta(w) + s\,\vartheta^1_\Delta(w)} \times \\
& \prod_{a\in A_1} \sum_{m_a,u_a} E_{m_a,u_a}\left[\exp\left\{\varepsilon_a\,\theta\,A^a_1\right\}\,\mathbf{1}(M^a_1 = m', U^a_1 = u')\right]\,\hat{a}^a_\theta(m'_a,u'_a) \times \\
& \prod_{a\in A_2} \sum_{m_a,u_a} E_{m_a,u_a}\left[\exp\left\{(\varepsilon_a\,\theta - s)\,A^a_1\right\}\,\mathbf{1}(M^a_1 = m', U^a_1 = u')\right]\,\hat{a}^{*a}_s(m'_a,u'_a) \times \\
& E_{v_1}\left[\exp\left\{(-s - d_1\,\theta)\,B^1_1\right\}\,\mathbf{1}(V^1_1 = v'_1)\right]\,\hat{a}^{*1}_{-s}(v'_1) \times \\
& E_{v_2}\left[\exp\left\{(s - d_2\,\theta)\,B^2_1\right\}\,\mathbf{1}(V^2_1 = v'_2)\right]\,\hat{a}^{*2}_s(v'_2) \\
= \;& e^{\theta\,\zeta_\Delta(w) - s\,\vartheta^2_\Delta(w) + s\,\vartheta^1_\Delta(w)} \times \prod_{a\in A_1} e^{-\alpha_a(\theta)}\,\hat{a}^a_\theta(m_a,u_a) \times \prod_{a\in A_2} e^{-\alpha^*_a(s)}\,\hat{a}^{*a}_s(m_a,u_a) \times \\
& e^{-\beta^*_1(-s)}\,\hat{a}^{*1}_{-s}(v'_1) \times e^{-\beta^*_2(s)}\,\hat{a}^{*2}_s(v_2).
\end{aligned}
$$

The last equality follows from Lemma 3.5.1, Lemma 3.4.3, and Lemma 3.6.1.

Since

$$\Lambda_2(s) = \sum_{a\in A_1} -\alpha_a(\theta) + \sum_{a\in A_2} -\alpha^*_a(s) - \beta^*_1(-s) - \beta^*_2(s),$$

then it follows that

$$\sum_{w'} K(w,w')\,e^{\theta\,(e'-e) - s\,(q'_2 - q_2)}\hat{a}^{*2}_s(\hat{w}') = e^{\theta\,\zeta_\Delta(w) - s\,\vartheta^2_\Delta(w) + s\,\vartheta^1_\Delta + \Lambda_2(s)}\,\hat{a}^{*2}_s(\hat{w}).$$

This proves the case (ii).

□

Appendix B

Markov Chains

In this chapter, we consider a discrete time irreducible Markov chain $X = \{X[n] : n \in \mathbb{Z}_+\}$, where $\mathbb{Z}_+$ is the set of non-negative integers, defined on the probability space $(\Omega, \mathcal{F}, \mathsf{P}_x)$ and evolving on the countable state space S. Its transition kernel is denoted K. We denote Δ as the drift operator

$$\Delta V(x) := \sum_{y \in S} K(x,y)V(y) - V(x), \quad x \in \mathsf{X},$$

where V is any non-negative (or bounded) measurable function,

B.1 Transience

Definition B.1.1 (Hitting Time) *Let $A \subset S$. Define τ_A as*

$$\tau_A := \min\{n \geq 1 : X[n] \in A\}.$$

Definition B.1.2 (Transient Set) *We will say that $A \subset S$ is a transient set, if for some $x \in S$*

$$\mathsf{P}_x(\tau_A = \infty) > 0.$$

We reformulate Theorem 3.6 from Chapter 5 in [Br99]. Therein, it is stated as a sufficient condition of transcience for a Markov chain. Actually, it is a sufficient condition of set transcience as defined above.

Theorem B.1.3 *Let A be a proper subset (possibly infinite) of S. Let $V : S \to \mathbb{R}$ be a bounded function such that*

$$\Delta V(x) \leq 0, \text{ for all } x \in S \setminus A. \tag{B.1.1}$$

Suppose, moreover, that there exists $x \in S \setminus A$ such that

$$V(x) < V(y), \text{ for all } y \in A. \tag{B.1.2}$$

Then, $\mathsf{P}_x(\tau_A = \infty) > 0$. Thus the set A is transient.

Proof : We utilize the following notation $\mathcal{F}_n = \sigma(X[0], \ldots, X[n])$. Let $x \in S \setminus A$ such that it satisfies (B.1.2). Define $Y[n] = V(X[n \wedge \tau_A])$. We have the following :

$$\begin{aligned}
\mathsf{E}_x\left[Y[n+1]\,|\mathcal{F}_n\right] &= \mathsf{E}_x\left[\mathbf{1}_{(n<\tau_A)}\,V(X[n+1])\,|\mathcal{F}_n\right] + \mathsf{E}_x\left[\mathbf{1}_{(n\geq\tau_A)}\,V(X[\tau_A])\,|\mathcal{F}_n\right] \\
&= \mathsf{E}_x\left[\mathbf{1}_{(n<\tau_A)}\,V(X[n+1])\,|\mathcal{F}_n\right] + \mathbf{1}_{(n\geq\tau_A)}\,V(X[\tau_A]) \\
&= \mathsf{E}_x\left[\mathbf{1}_{(n<\tau_A)}\,V(X[n+1])\,|\mathcal{F}_n\right] + \mathbf{1}_{(n\geq\tau_A)}\,Y[n].
\end{aligned}$$

Using (B.1.1), we obtain the following inequality

$$\mathsf{E}_x\left[\mathbf{1}_{(n<\tau_A)}\,V(X[n+1])\,|\mathcal{F}_n\right] \leq \mathbf{1}_{(n<\tau_A)}\,V(X[n]) = \mathbf{1}_{(n<\tau_A)}\,Y[n].$$

Therefore,

$$\mathsf{E}_x\left[Y[n+1]\,|\mathcal{F}_n\right] \leq Y[n].$$

Which means that Y is a bounded supermartingale with respect to the filtering $\mathcal{F}_n$. It is bounded since V is assumed bounded.

By the martingale convergence theorem, $Y[\infty] = \lim_{n\to\infty} Y[n]$ exists and is P_x-a.s. finite. By bounded convergence

$$\mathsf{E}_x[Y[\infty]] = \lim_{n\to\infty} \mathsf{E}_x[Y[n]],$$

and, since Y is a supermartingale, then

$$\mathsf{E}_x[Y[n]] \leq \mathsf{E}_x[Y[0]] = V(x).$$

Thus,

$$\mathsf{E}_x[Y[\infty]] \leq V(x). \tag{B.1.3}$$

Suppose the contrary that τ_A is P_x-a.s. finite, then $Y[n]$ would eventually be frozen at a value $V(y)$ for $y \in A$. This contradicts (B.1.3) since by (B.1.2) $V(x) < V(y)$ for all $y \in A$. Therefore, $\mathsf{P}_x(\tau_A = \infty) > 0$.

□

Definition B.1.4 (V-Transient Set) *We will say that $A \subset S$ is a V-transient set, if there exists a function V that satisfies the conditions of Theorem B.1.3.*

Appendix C

C++ Programs Using SimS

C.1 Source Code for IS

```
#include <fstream.h> #include <stdlib.h>    // for atof, atoi
#include <sims.h> #include <stdio.h>    // for sprintf #include
<string.h>   // for strcmp #include <new.h> #include "prog.h"
#include "buffer.h" #include "cell.h"

int NumReplications, SimTime, b_int,b_max, B[8]; int maxCTD,
NumCycles,Warmup;

void Multi_set_parameters() {

     NumReplications=atoi(sim_config()->get_parameter("num_replications"));
     NumCycles=atoi(sim_config()->get_parameter("num_cycles"));
     SimTime=atoi(sim_config()->get_parameter("sim_time"));
     b_int=atoi(sim_config()->get_parameter("buffer_int"));
     b_max=atoi(sim_config()->get_parameter("buffer_max"));
     B[0]=atoi(sim_config()->get_parameter("b1"));
     B[1]=atoi(sim_config()->get_parameter("b2"));
     B[2]=atoi(sim_config()->get_parameter("b3"));
     B[3]=atoi(sim_config()->get_parameter("b4"));
     B[4]=atoi(sim_config()->get_parameter("b5"));
     B[5]=atoi(sim_config()->get_parameter("b6"));
     B[6]=atoi(sim_config()->get_parameter("b7"));
     B[7]=atoi(sim_config()->get_parameter("b8"));
     maxCTD=atoi(sim_config()->get_parameter("maxctd"));
     Warmup=atoi(sim_config()->get_parameter("warmup"));

}

class Multi_config_obj:public sim_config_obj { public:
     Multi_config_obj()
     {
       add_parameter("num_replications","0");
       add_parameter("num_cycles","0");
       add_parameter("sim_time","0");
       add_parameter("buffer_int","0");
       add_parameter("buffer_max","0");
       add_parameter("b1","0");
       add_parameter("b2","0");
       add_parameter("b3","0");
```

```
    add_parameter("b4","0");
    add_parameter("b5","0");
    add_parameter("b6","0");
    add_parameter("b7","0");
    add_parameter("b8","0");
    add_parameter("maxctd","0");
    add_parameter("warmup","0");
  }
};

int main(int argc, char *argv[]) {

    char ConfigFileName[80];

    // make sure config file is an argument
    if (argc<2)
          { cout<<endl<<"the configuration file name :";
              cin>>ConfigFileName;

          }
      else { strcpy(ConfigFileName,argv[1]);
           }

    // get values of the parameters
    sim_set_config(new Multi_config_obj());
    sim_config()->read_file(ConfigFileName);
    Multi_set_parameters();


    // construct a queue object
    buffer_entity *Queue_ptr;


Queue_ptr = new buffer_entity("buffer",B[0],
     B[1], B[2], B[3],B[4],B[5],B[6],B[7],b_max,maxCTD);

cout<<"Start of Simulation"<<endl;

int r,j,c;
double Ratio_Rep_Bins[NumReplications][8], Ratio_Rep_Bins_CMC[NumReplications][8];
double sumDelays[8], sumCells, sumDelaysCMC[8];
double ISweight=1.0;
double AvgHitEnc=0.0; double CountCycles=0.0;
double AvgHitDelay[8], AvgHitDelayCMC[8];
double Delays_Rep_bins[NumReplications][8];
double Delays_Rep_bins_CMC[NumReplications][8];
double CountCells_Rep[NumReplications];

for (j=0;j<=7;j++) AvgHitDelay[j]=0.0;

for (r=1;r<=NumReplications;r++)
   {
cout<<" replication "<<r<<endl;
     // initialize the sim objects for a new replication
     Queue_ptr->initialize_replication();
     for (j=0;j<=7;j++)
        { sumDelays[j]=0.0; sumDelaysCMC[j]=0.0; }

     sumCells=0.0;

     for (c=1;c<=NumCycles+Warmup;c++)
     {
        // initialize for phase 1 - twisted cycle

        if (c>Warmup)
        {

        CountCycles=CountCycles+1.0;

        Queue_ptr->init_twist();
        sim_run_simulation();
        Queue_ptr->Suspend();

        ISweight=Queue_ptr->LR;

        AvgHitEnc=AvgHitEnc*(CountCycles-1.0);
        for (j=0;j<=7;j++)
        {
            AvgHitDelay[j]=AvgHitDelay[j]*(CountCycles-1.0);
            AvgHitDelayCMC[j]=AvgHitDelayCMC[j]*(CountCycles-1.0);
        }
```

```
                AvgHitDelayCMC[j]=AvgHitDelayCMC[j]/CountCycles;
                //   cout<<" CMC delay "<<j<<"  prop "<<AvgHitDelayCMC[j]<<endl;
            }

            if (Queue_ptr->hit)
            {

                // initialize to end cycle untwisted
                Queue_ptr->untwist();
                sim_run_simulation();
                Queue_ptr->Suspend();
                AvgHitEnc=AvgHitEnc+1.0;
                                                    sumCells=sumCells+Queue_ptr->CountCells;
                                                }
                                                }
            }
                                                for (j=0;j<=7;j++)
                                                {
                AvgHitEnc=AvgHitEnc/CountCycles;                    Ratio_Rep_Bins[r-1][j]=sumDelays[j]/sumCells;
//   cout<<" hit enc "<<AvgHitEnc<<endl;                cout<<" bin "<<j<<"  ratio "<<sumDelays[j]/sumCells<<endl;
                if (c>Warmup)                                       cout<<"   # delays "<<sumDelays[j]<<"  # cells "<<sumCells<<endl;
                {                                                   Ratio_Rep_Bins_CMC[r-1][j]=sumDelaysCMC[j]/sumCells;

                    for (j=0;j<=7;j++)
                    {                                               Delays_Rep_bins[r-1][j]=sumDelays[j];
                        sumDelays[j]=sumDelays[j]+(ISweight*Queue_ptr->BinsExtreme[j]);
                        if (Queue_ptr->BinsExtreme[j]>0)            Delays_Rep_bins_CMC[r-1][j]=sumDelaysCMC[j];
                        {                                               }
                            AvgHitDelay[j]=AvgHitDelay[j]+1.0; }        CountCells_Rep[r-1]=sumCells;
                            AvgHitDelay[j]=AvgHitDelay[j]/CountCycles;  }
                            //  cout<<" delay "<<j<<"  prop "<<AvgHitDelay[j]<<endl;
                    }                                           cout<<"End of Simulation"<<endl;
                }                                               cout<<endl;
            }

            // initialize for phase 2 - CMC                     int k;
                                                                double sum[8], SS[8], numReps=(double)NumReplications;
            Queue_ptr->initialize_CMC();                        double avg[8],sampleVar[8],sampleSt_dev[8], RE[8];
            sim_run_simulation();                               numReps=numReps-1.0;
            Queue_ptr->Suspend();
                                                                for (j=0;j<=7;j++)
            if (c>Warmup)                                       {
            {                                                   sum[j]=0.0;
                for (k=1;k<NumReplications;k++)
                {                                               {
                    sum[j]=sum[j]+Ratio_Rep_Bins[k][j];
                    SS[j]=SS[j]+Ratio_Rep_Bins[k][j]*Ratio_Rep_Bins[k][j];
                }
                avg[j]=sum[j]/numReps;
                sampleVar[j]=(numReps*SS[j]-(sum[j]*sum[j]))/(numReps*(numReps-1.0));
                sampleSt_dev[j]=sqrt(sampleVar[j]);
                RE[j]=2.57*sampleSt_dev[j]/(avg[j]*sqrt(numReps));
```

```
                sumDelaysCMC[j]=sumDelaysCMC[j]+Queue_ptr->BinsCMC[j];
                if (Queue_ptr->BinsCMC[j]>0)
                {
                    AvgHitDelayCMC[j]=AvgHitDelayCMC[j]+1.0; }
```

```
    }

    double sumCMC[8], SSCMC[8];
    double avgCMC[8],sampleVarCMC[8],sampleSt_devCMC[8], RECMC[8];


    for (j=0;j<=7;j++)
      {
        sumCMC[j]=0.0;
        for (k=1;k<NumReplications;k++)
      {
          sumCMC[j]=sumCMC[j]+Ratio_Rep_Bins_CMC[k][j];
          SSCMC[j]=SSCMC[j]+Ratio_Rep_Bins_CMC[k][j]*Ratio_Rep_Bins_CMC[k][j];
      }
        avgCMC[j]=sumCMC[j]/numReps;
        sampleVarCMC[j]=(numReps*SSCMC[j]-(sumCMC[j]*sumCMC[j]))/(numReps*(numReps-1.0));
        sampleSt_devCMC[j]=sqrt(sampleVarCMC[j]);
        RECMC[j]=2.57*sampleSt_devCMC[j]*sqrt(numReps));


    }


    // estimator is Y/X
    double sumY[8], SSY[8], sumX, SSX, SSXY[8];
    double avgY[8], avgX;
    double sampleVarY[8], sampleVarX, sampleCovXY[8];
    double mean[9], var[8], std[8], re[8];


    sumX=0.0;
    SSX=0;
    for (k=1;k<NumReplications;k++)
      {
        sumX=sumX+CountCells_Rep[k];
        SSX=SSX+CountCells_Rep[k]*CountCells_Rep[k];
      }
    avgX=sumX/numReps;
    sampleVarX=(numReps*SSX-(sumX*sumX))/(numReps*(numReps-1.0));


    for (j=0;j<=7;j++)
    {
      sumY[j]=0.0;
      SSY[j]=0.0;
      SSXY[j]=0.0;
      for (k=1;k<NumReplications;k++)
        {
          sumY[j]=sumY[j]+Delays_Rep_bins[k][j];
          SSY[j]=SSY[j]+Delays_Rep_bins[k][j]*Delays_Rep_bins[k][j];
          SSXY[j]=SSXY[j]+Delays_Rep_bins[k][j]*CountCells_Rep[k];
        }
      avgY[j]=sumY[j]/numReps;
      sampleVarY[j]=(numReps*SSY[j]-(sumY[j]*sumY[j]))/(numReps*(numReps-1.0));
      sampleCovXY[j]=(SSXY[j]-numReps*avgX*avgY[j])/(numReps-1.0);

      mean[j]=avgY[j]/avgX;
      var[j]=(sampleVarY[j]+mean[j]*mean[j]*sampleVarX-2*mean[j]*sampleCovXY[j])/
                                    (numReps*avgX*avgX);
      std[j]=sqrt(var[j]);
      re[j]=2.57*std[j]/mean[j];

    }

    cout<<"# of replications "<<NumReplications<<endl;

    cout<<endl<<endl;

    for (j=0;j<=7;j++)
      {
        cout<<"j="<<B[j]<<"   avg  (IS)  "<<avg[j]<<" pm "<<RE[j];
        cout<<"   avg  (CMC) "<<avgCMC[j]<<" pm "<<RECMC[j];
        cout<<"   avg  (alt IS)  "<<mean[j]<<" pm "<<re[j]<<endl;
      }

    for (j=0;j<=7;j++)
      {
        cout<<"j="<<B[j]<<"   prop. hit    (IS) "<<AvgHitDelay[j];
        cout<<"   (CMC)  "<<AvgHitDelayCMC[j]<<endl;
      }

    cout<<" prop. Hit ENC (IS) "<<AvgHitEnc<<endl;
```

```
    system("ps -e |grep prog");

}

#include "buffer.h"


buffer_entity::buffer_entity(const char *name, int bin1,
        int bin2, int bin3, int bin4, int bin5, int bin6, int bin7,int bin8,
            int max, int ctd):sim_entity(name)
{
    strcpy(Name,name);

    maxSize=max;
    rep=-1;


    p10=0.3333333333; p11=1-p10;
    p01=0.0166666667; p00=1-p01;
    InterArrival=3;

    int k;

    NumOfSources=8;
    NumOfBuffers=2;


    for (k=0;k<NumOfSources;k++)
      {
        ExitAt[k]=0;
        if (k<=2) {
                SourceEnc[k]=5;
                OfferToBuffer[k]=1;
                p01Twisted[k] = 0.04905979281;
                p10Twisted[k] = 0.1410939860;
            }
        else { SourceEnc[k]=2;
               OfferToBuffer[k]=0;
               p01Twisted[k] = 0.0247087807;
               p10Twisted[k] = 0.2506577846;
             }
      }
}

for (k=0;k<NumOfSources;k++)
  {
    p11Twisted[k]=1-p10Twisted[k];
    p00Twisted[k]=1-p01Twisted[k];
  }

int i;
for (i=0;i<=1;i++)
  {
     sprintf(Queuename[i],"%s%d",name,i);
     create_message_queue(Queuename[i]);
     Queue[i]=find_queue(Queuename[i]);
     Empty[i]=TRUE;
     sprintf(tQueuename[i],"twist%s%d",name,i);
     create_message_queue(tQueuename[i]);
     tQueue[i]=find_queue(tQueuename[i]);
     tEmpty[i]=TRUE;
  }

for (i=0;i<NumOfSources;i++)
  {  CurrentState[i]=0;
     ResidualTime[i]=0;
     tCurrentState[i]=0;
     tResidualTime[i]=0;
  }

CTD=ctd;

level[0]=bin1;
level[1]=bin2;
level[2]=bin3;
level[3]=bin4;
level[4]=bin5;
level[5]=bin6;
level[6]=bin7;
level[7]=bin8;


for (i=0;i<=7;i++)
  { BinsExtreme[i]=0; }

NextBuffer[0]=2;
NextBuffer[1]=0;

InterService[0]=2;
InterService[1]=3;
```

```
    // create a Random Number Gen. object
     pRandom = new sim_generator();

}


buffer_entity::~buffer_entity() {}


void buffer_entity::execute() {

     int i,j;
     sim_bool bern_rv;
     cell *pCell;
     int delay;

     double arrivalrate=0.0;
     double timecount=0.0;

     while (TRUE)
    {

     time++;


     if (cmc)
       {

          // arrivals and state transitions for sources
          for (i=0;i<NumOfSources;i++)
           {
             if (CurrentState[i]==1)
           { // source is on

                 if (ResidualTime[i]==0)
            {
                      if (Queue[OfferToBuffer[i]]->length()<maxSize)
                {
                     pCell = new cell(time,i, SourceEnc[i], ExitAt[i]);
                     enc=enc+SourceEnc[i];

                     send_message(pCell,Queuename[OfferToBuffer[i]]);


                }
                pRandom->draw(bern_rv, p10);
                if (bern_rv) { CurrentState[i]=0;
                 }
               else { ResidualTime[i]=InterArrival-1; }
   } else ResidualTime[i]--;

  } else { // source is off

               pRandom->draw(bern_rv, p01);
               if (bern_rv) { CurrentState[i]=1;
                             ResidualTime[i]=InterArrival-1;
                    }
         }
     } // end for i=0 to NumOfSources


  if (ResidualTimeBuffer[0]==0)
{
       if (!Empty[0])
         {
          // serve cell from Buffer 0
          pCell=(cell *) retrieve(NO_WAIT,IGNORE_ACTIVATES,Queue[0]);
          if (pCell->ID()==0)
    {
               CountCells=CountCells+1.0;
               delay=time-pCell->time_offered();
               for (j=0;j<=7;j++)
      {
                    if ( delay> level[j] ) BinsCMC[j]=BinsCMC[j]+1.0;
      }
     }
          enc=enc-InterService[0];
          delete (pCell);
      }
        ResidualTimeBuffer[0]=InterService[0]-1;
} else ResidualTimeBuffer[0]--;



for (i=1;i<NumOfBuffers;i++)
   {
     if (ResidualTimeBuffer[i]==0)
   {
          if (!Empty[i])
            {
             // serve cell from Buffer i
             pCell=(cell *) retrieve(NO_WAIT,IGNORE_ACTIVATES,Queue[i]);
```

```
        if (pCell->exit()==i)
          {
            delete(pCell);
          if (i==0||i==1) enc=enc-InterService[i];
          }
        else {   if (Queue[NextBuffer[i]]->length()<maxSize)
                  {
                    if (i==0||i==1)
                    {
                        pCell->decrement_enc(InterService[i]);
                        enc=enc-InterService[i];
                    }
                    send_message(pCell,Queuename[NextBuffer[i]]);
                  }
                  else {
                        enc=enc-pCell->get_enc();
                        delete (pCell);
                  }
              }
          }

        ResidualTimeBuffer[i]=InterService[i]-1;

    } else { ResidualTimeBuffer[i]--; }
    }


    for (i=0;i<NumOfBuffers;i++)
      {
       Empty[i]=(Queue[i]->length()==0);
      }

    if (Empty[0]&&Empty[1]) stop=TRUE;

    } // end cmc
    else {  if (twist)
              {


timecount=timecount+1.0;

            // arrivals and state transitions for twisted sources
            for (i=0;i<NumOfSources;i++)
              {
                if (tCurrentState[i]==1)
                { // twisted source is on

                    if (tResidualTime[i]==0)
                    {
arrivalrate=arrivalrate+1.0;
                        if (tQueue[OfferToBuffer[i]]->length()<maxSize)
                        {
                            pCell = new cell(time,i, SourceEnc[i],ExitAt[i]);
                            send_message(pCell,tQueuename[OfferToBuffer[i]]);

                            enc=enc+SourceEnc[i];
                        }
                        pRandom->draw(bern_rv, p10Twisted[i]);

                        if (bern_rv) { tCurrentState[i]=0;
                                       LR=LR*p10/p10Twisted[i];
                                     } else { LR=LR*p11/p11Twisted[i];
                                              tResidualTime[i]=InterArrival-1; }
                    } else tResidualTime[i]--;

                } else { // source is off
                            pRandom->draw(bern_rv, p01Twisted[i]);
                            if (bern_rv) { tCurrentState[i]=1;
                            LR=LR*p01/p01Twisted[i];
                                    tResidualTime[i]=InterArrival-1;
                            } else { LR=LR*p00/p00Twisted[i]; }
                       }

                } // end for i=0 to NumOfSources


            if (tResidualTimeBuffer[0]==0)
            {
              if (!tEmpty[0])
              {
                // serve cell from tBuffer 0
                pCell=(cell *) retrieve(NO_WAIT,IGNORE_ACTIVATES,tQueue[0]);
                if (pCell->ID()==0)
                {
                    delay=time-pCell->time_offered();

                    for (j=0;j<=7;j++)
                    {
                        if (delay>level[j]) BinsExtreme[j]=BinsExtreme[j]+1.0;
                    }
```

```
}
                                        } else { tResidualTimeBuffer[i]--; }

                                    }

        enc=enc-InterService[0];

          delete (pCell);
      }
      tResidualTimeBuffer[0]=InterService[0]-1;

                                  for (j=0;j<NumOfBuffers;j++)
                                  {
                                    tEmpty[j]=(tQueue[j]->length()==0);
                                  }

  } else tResidualTimeBuffer[0]--;

                                  if (enc>=CTD) { stop=TRUE; hit=TRUE; }
                                  if (tEmpty[0]&&tEmpty[1]) stop=TRUE;

                                } // end twist
                                  else { // end the twisted cycle untwisted

                                      // arrivals and state transitions for twisted sources
                                      for (i=0;i<NumOfSources;i++)
                                        {
                                        if (tResidualTime[i]==0)
                {                       {
                                          if (tCurrentState[i]==1)
                                        { // twisted source is on

                                            if (tQueue[OfferToBuffer[i]]->length()<maxSize)
                                            {
                                              pCell = new cell(time,i, SourceEnc[i], ExitAt[i]);
                                              enc=enc+SourceEnc[i];
                                              send_message(pCell,tQueuename[OfferToBuffer[i]]);
                                            }
                                            pRandom->draw(bern_rv, p10);
                                            if (bern_rv) { tCurrentState[i]=0; }
                                          else { tResidualTime[i]=InterArrival-1; }
                                          } else { // source is off
                                                   pRandom->draw(bern_rv, p01);
                                                   if (bern_rv) {
                                                                  tCurrentState[i]=1;
                                                                  tResidualTime[i]=InterArrival-1;
                                                   }
                                                   }
                                        } else { tResidualTime[i]--; }
                                      } // end for i=0 to NumOfSources

  for (i=1;i<NumOfBuffers;i++)
    {
      if (tResidualTimeBuffer[i]==0)
      {
          if (!tEmpty[i])
            {
              // serve cell from Buffer i
              pCell=(cell *) retrieve(NO_WAIT,IGNORE_ACTIVATES,tQueue[i]);
              if (pCell->exit()==i)
                  { delete(pCell);
                    if (i==0||i==1) { enc=enc-InterService[i];
                                    }
                  }
              else {   if (tQueue[NextBuffer[i]]->length()<maxSize)
                       {
                            if (i==0||i==1)
                              {
                                enc=enc-InterService[i];
                              }
                            pCell->decrement_enc(InterService[i]);
                       }
                   send_message(pCell,tQueuename[NextBuffer[i]]);
                  }
            }
          else  { delete (pCell);
                  enc=enc-pCell->get_enc();
                }
      }
    }
  }

  tResidualTimeBuffer[i]=InterService[i]-1;
```

```
                                                    }
                                                    else  { delete (pCell);
                                                            enc=enc-pCell->get_enc();
                                                          }
                                                      }
                                                    }
                                                }
                                                tResidualTimeBuffer[i]=InterService[i]-1;
                                                } else { tResidualTimeBuffer[i]--; }
                                            }
                                            for (i=0;i<NumOfBuffers;i++)
                                            {
                                                tEmpty[i]=(tQueue[i]->length()==0);
                                            }
                                            if (tEmpty[0]&&tEmpty[1]) stop=TRUE;

                                                } // !twist

                                            } //end !cmc

                                    if (stop) sim_end_simulation();
                                      else activate_in(1.0);
                                }
                            void buffer_entity::initialize_replication() {
                                int i;
                                for (i=0;i<NumOfBuffers;i++)
                                    {
                                        Queue[i]->clear();
                                        tQueue[i]->clear();
                                        Empty[i]=TRUE;
                                        tEmpty[i]=TRUE;
                                        ResidualTimeBuffer[i]=0;
if (tResidualTimeBuffer[0]==0)
{
    if (!tEmpty[0])
    {
        // serve cell from tBuffer 0
        pCell=(cell *) retrieve(NO_WAIT,IGNORE_ACTIVATES,tQueue[0]);
        if (pCell->ID()==0)
        {
            delay=time-pCell->time_offered();
            for (j=0;j<7;j++)
            {
                if (delay>level[j]) BinsExtreme[j]=BinsExtreme[j]+1.0;
            }
        }
        enc=enc-InterService[0];
        delete (pCell);
    }
    tResidualTimeBuffer[0]=InterService[0]-1;
} else tResidualTimeBuffer[0]--;

    for (i=1;i<NumOfBuffers;i++)
    {
        if (tResidualTimeBuffer[i]==0)
        {
            if (!tEmpty[i])
            {
                // serve cell from Buffer i
                pCell=(cell *) retrieve(NO_WAIT,IGNORE_ACTIVATES,tQueue[i]);
                if (pCell->exit()==i)
                    { delete(pCell);
                      if (i==0||i==1) enc=enc-InterService[i];
                    }
                else {  if (tQueue[NextBuffer[i]]->length()<maxSize)
                        {
                            if (i==0||i==1)
                            {
                                enc=enc-InterService[i];
                                pCell->decrement_enc(InterService[i]);
                            }
                            send_message(pCell,tQueuename[NextBuffer[i]]);
```

```
        tResidualTimeBuffer[i]=0;
        else { tEmpty[i]=TRUE; }
      }
      for (i=0;i<NumOfSources;i++)
        {
          CurrentState[i]=0;
          ResidualTime[i]=0;
        }
      time=0;
  }
  void buffer_entity::init_twist() {
    enc=0;
    stop=FALSE;
    cmc=FALSE;
    hit=FALSE;
    TimeBeforeTwist=time;
    int NumToCopy;
    iR=1.0;
    cell *pCell, *pCell_Copy;
    int i,j;
    for (i=0;i<NumOfBuffers;i++)
      {
        tQueue[i]->clear();
        NumToCopy=Queue[i]->length();
        if (NumToCopy>0)
        {
          for (j=1;j<=NumToCopy;j++)
            {
              pCell=(cell*) peek_kth(j,Queue[i]);
              pCell_Copy = new cell(pCell->time_offered(),
                  pCell->ID(), pCell->get_enc(),pCell->exit());
              send_message(pCell_Copy,tQueuename[i]);
        }
        tEmpty[i]=FALSE;
      }
  }
      for (i=0;i<NumOfSources;i++)
        {
          tCurrentState[i]=CurrentState[i];
          tResidualTime[i]=ResidualTime[i];
        }
      for (i=0;i<=7;i++)
        {
          BinsExtreme[i]=0.0;
        }
      activate_in(1.0);
  }
  void buffer_entity::untwist() {
    stop=FALSE;
    cmc=FALSE;
    twist=FALSE;
    activate_in(1.0);
  }
  void buffer_entity::initialize_CMC() {
    stop=FALSE;
    cmc=TRUE;
    CountCells=0.0;
    enc=0;
```

```
    time=TimeBeforeTwist;

    CountCells=0.0;
    int i;
    for (i=0;i<=7;i++)
      {
        BinsCMC[i]=0.0;
      }

    activate_in(1.0);
}

void buffer_entity::Suspend() {
 suspend();
}

#include "cell.h"

cell::cell(int time, int ID,int ENC, int exit) {
    offered = time;
    id =ID;
    enc=ENC;
    Exit=exit;
} cell::~cell() {}

const char *cell::type() const { return ("cell"); }

int cell::time_offered() {
  return (offered);
}

int cell::ID() {
  return (id);
}

int cell::get_enc() {
  return (enc);
}

}

int cell::exit() {
  return (Exit);
}

void cell::decrement_enc(int ENC) {

    enc=enc-ENC;

}

#ifndef CellH #define CellH

#include <sims.h> #include "prog.h" #include <iostream.h>  //
extra for debugging #include <math.h>

class cell: public sim_message { public:
    cell(int, int, int,int);
    ~cell();
    const char *type() const;

    void set_time(double time);
    void decrement_enc(int ENC);
    int time_offered();
    int ID();
    int get_enc();
    int exit();
  private:
    int offered;
    int id;
    int enc;
    int Exit;

  };

  #endif
```

Bibliography

[Asmu87] Asmussen, S. (1987) *Applied Probability and Queues*, John Wiley & Sons.

[BDM99] Beck B., Dabrowski A.R., and McDonald D.R. (1999) *A Unified Approach to Fast Teller Queues and ATM*, Adv. Appl. Prob. **31**, 758–787.

[Bonn96] Bonneau, M-C. (1996) *Accelerated Simulation of a Leaky Bucket Controller*, M.Sc. thesis, University of Ottawa.

[Br99] Brémaud, Pierre (1999) *Markov Chains : Gibbs Fiels, Monte Carlo Simulation and Queues*, Springer.

[BFS87] Bratley P., B. Fox, and L. Schrage (1987) *A Guide to Simulation*, Springer

[CHJS94] Chang C.-S., Heidelberger P., Juneja., and Shahabuddin P. (1994) *Effective Bandwidth and Fast Simulation of ATM intree Networks*, Performance Evaluation, **20**, 45–65.

[CT01] Corcoran, J., and Tweedie, R. (2001) *Perfect Sampling of Ergodic Harris Chains*, Ann. App. Prob., **11**, no. 2, 438-451.

[DLM00] Dabrowski, A.R., Lamothe, G., and McDonald D.R. (2000). *Accelerated Simulation of ATM Switching Fabrics*, Fields Institute Communications, **26**, 193–205.

[Dai95] Dai, J.G. (1995). *On Positive Harris Recurrence of Multiclass Queueing Networks: A Unified Approach via Fluid Limit Models*, Ann. Appl. Probab., **5**, no. 1, 49-77.

[DO98] Dembo, A. and Zeitouni, O. (1998) *Large Deviations Techniques and Applications*, 2nd edition, Springer-Verlag.

[DM97] Down, D. and S.P. Meyn. (1997) *Piecewise linear test functions for stability and instability of queueing networks*, Queueing Systems: Theory and Applications, **27**, pp. 205-226.

[FDL99] Falkner M., Devetsikiotis M., and Lambadaris L. (1999) *Fast Simulation of Networks of Queues by Use of Effective and Decoupling Bandwidths*, ACM Trans. Model. Comput. Simul., **9**, 1, 45–58.

[F98] Fill, J.A. (1998). *An interruptible algorithm for perfect sampling via Markov Chains*, Fields Institute Communications, **8**, no. 1, 131–162.

[FM03] Foley, R.D. and McDonald, D.R. (2003) *Join the Shortest Queue: Stability and Exact Asymptotics*, to appear, Annals of Applied Probability.

[GO98] Ganesh, A.J. and O'Connell, N. (1998) *The Linear Geodesic Property is Not Generally Preserved by a FIFO queue*, Annals of Applied Probability, **8**, no. 1, 98-111.

[GHSZ96] Glasserman, P., Heidelberger, P., Shahabuddin, P., and Zajic, T. (1996) *Splitting for Rare Events Simulation: Analysis of Simple Cases*, Proceedings of the 1996 Winter Simulation Conference, IEEE Press (1996), 302–308.

[GK95] Glasserman, P., and Kou, S.-K. (1995) *Analysis of an Importance Sampling Estimator for Tandem Queues*, ACM Trans. Model. Comput. Simul., **5**, 1, 22–42.

[GSHNG92] Goyal, A., Shahabuddin, P., Heidelberger, P., Nicola, V.F. and Glynn, P.W. (1992) *A Unified Framework for Simulating Markovian Models of Highly Dependable Systems*, IEEE Transactions on Computers, **41**,1, 36–51.

[GH90] Grassmann, W.K. and Heyman, D.P. (1990) *Equilibrium distribution of block-structured Markov chains with repeated rows*, J. Appl. Prob., **27**, 557-576.

[GR65] Gunning, R.C., and Rossi, H. (1965) *Analytic Functions of Several Complex Variables*, Prentice-Hall.

[Heid95] Heidelberger P. (1995) *Fast Simulation of Rare Events in Queueing and Reliability Models*, ACM Transactions on Modeling and Computer Simulation, **5**, 43–85.

[KSK76] Kemeny, J.G., Snell, J.L. and Knapp, A.W. (1976) *Denumerable Markov Chains*, Springer-Verlag.

[K97] Knuth, D. E. (1997) *Seminumerical Algorithms*, Addison-Wesley

[KN02] Kroese, D.P., and V.F. Nicola *Efficient Simulation of a Tandem Jackson Network*, ACM Trans. on Modeling and Computer Simulation (TOMACS), **12**, no. 2, 119-141.

[LK00] Law, A.M., and Kelton, D.W. (2000) *Simulation Modeling and Analysis*, McGraw-Hill

[L88] L'Ecuyer, P. (1988) *Efficient and Portable Combined Random Number Generators*, Communications of the ACM, **31** 742–749.

[LC96] L'Ecuyer, P., and Champoux, Y. (1996) *Importance Sampling for Large ATM-Type Queueing Networks*, Proceeding of 1996 Winter Simulation Conference, IEEE Press, 309–316.

[LC01] L'Ecuyer, P., and Champoux, Y. (2001) *Estimating Small Cell-Loss Ratios in ATM Switches via Importance Sampling*, ACM Transactions on Modeling and Computer Simulation, **11**, no. 1, .

[McDo99] McDonald, D.R. (1999) *Asymptotics of First Passage Times for Random Walk in an Orthant*, Ann. Appl. Probab., **9**, no. 1, 100–145.

[MT93] Meyn, S.P. and Tweedie, R.L. (1993) *Markov Chains and Stochastic Stability*, Springer Verlag.

[MT94] Meyn, S.P. and Tweedie, R.L. (1994) *State-Dependent Criteria for Convergence of Markov Chains*, Ann. Appl. Probab., **4**, no. 1, 149–168.

[NN87] Ney P., and Nummelin E. (1987) *Markov Additive Processes I. Eigenvalue Properties and Limit Theorems*, Ann. of Probab., **15**, no. 2, 561-592.

[PW89] Parekh, S. and Walrand, J. (1989) *A Quick Simulation Method for Excessive Backlogs in Networks of Queues*, IEEE Transactions on Automatic Control, **34**, 54–66.

[PW96] Propp, J.W. and Wilson, D.B. (1996) *Exact sampling with coupled Markov chains and applications to statistical mechanics*, Random Structures and Algorithms, **9**, 223–252.

[Q56] Quenouille, M. H. (1956) *Notes on bias in estimation*, Biometrika, **43**, 353–360.

[R02] Rolls, D.A. (2002) *Limit Theorems and estimation for structural and aggregate teletraffic models*, Ph.D. thesis, Queen's University, Canada.

[RS92] Rybko, A.N. and Stolyar, A.L. (1992) *Ergodicity of Stochastic Processes Describing the Operation of Open Queuing Networks*, Problems Inform. Transmission, **28**, 199–220.

[ATM96] *Traffic Management Specification Version 4.0*, ATM Forum, April 1996.

[TC00] TWEEDIE, R.L. AND CORCORAN, J.N. (2000). Perfect sampling and queuing models. To appear in *Proc. 38th Annual Allerton Conference on Communication, Control, and Computing*

[WF93] Wang, Q. and Frost, V.S. (1993) *Efficient Estimation of Cell Blocking Probability for ATM Systems*, ACM Transactions on Networking, **1**, 2, 230–235.

Glossary of Symbols

Glossary of Symbols - Perfect Sampling

Index