Noé Falzon

Optimal inter-cell coordination for elastic data traffic

Faculty of Electronics, Communications and Automation

Thesis submitted for examination for the degree of Master of Science in Technology.

Espoo 25.08.2010

Thesis supervisor:

Prof. Samuli Aalto

Thesis instructor:

D.Sc. (Tech.) Pasi Lassila

Aalto University
School of Science
and Technology

AALTO UNIVERSITY
SCHOOL OF SCIENCE AND TECHNOLOGY

ABSTRACT OF THE
MASTER'S THESIS

Author: Noé Falzon

Title: Optimal inter-cell coordination for elastic data traffic

Date: 25.08.2010 Language: English Number of pages:6+74

Faculty of Electronics, Communications and Automation
Department of Communications and Networking
Professorship: Networking Technology Code: S-38
Supervisor: Prof. Samuli Aalto
Instructor: D.Sc. (Tech.) Pasi Lassila

In cellular networks, where the wireless access is fairly shared among users, individual data rates are subject to variation because of the random nature of the number of active flows in a cell. Another factor is the radio channel quality, which depends highly on the users' positions in the cell, and on the interference caused by nearby base stations.
While user activity is random, its effects can be compensated by inter-cell interference coordination. This thesis proposes a queuing model of two interfering base stations, and considers the possibility of turning them on and off depending on the number of users in each cell.
We then devise an optimal time scheduling policy, that is a set of rules that minimizes the average total number of users in the system and therefore the average delay.
We establish that depending on the level of interference, it is beneficial to either serve always both cells at the same time, or on the contrary, to always serve only one queue at a time.

Keywords: Elastic data traffic, interference mitigation, inter-cell coordination, optimal time scheduling, Markov decision process, policy improvement

Preface

This thesis represents seven months of work as a research assistant in the department of Communications and Networking of Aalto University's School of Science and Technology. I would like to thank the persons who made this achievement possible.

The first word goes of course to Samuli Aalto for his interest in my curriculum, and who was kind enough to offer me the position in the laboratory; followed closely by Pasi Lassila for his friendly supervision and always pertinent advice.

Must be mentioned the laboratory's staff, for their great handling of administrative matters: Arja Hänninen, Sanna Patana, Sari Kiveliö, Lotta Timonen and Hanna Ropponen; Viktor Nässi for his warm welcome and cheerful presence; and Kimmo Pitkäniemi for his mastery of the arcane art of IT support.

Last but not least, my entire studies at the university would have been quite a different experience without the availability and unconditional support of the office of international affairs of the faculty: Anita Bisi and Jenni Tulensalo.

Otaniemi, 25.08.2010

Noé Falzon

Contents

Symbols and abbreviations

Mathematical notation

Unless explicitely stated otherwise, the following typefaces are used to denote the type of mathematical object:

$\mathbf{M}$	a matrix
m_{ij} or $m_{i,j}$	component on the i-th row and j-th column of matrix $\mathbf{M}$
$\mathbf{v}$	a column vector
v_i	i-th component of vector $\mathbf{v}$
$\bar{X} = E[X]$	the average or expectation of the random variable X
$P[X < x]$	the probability that the random variable X takes a value lower than x
$\lg(x)$	the base-2 logarithm, used for information theory purposes

Note that with the convention of using column vectors, the only valid matrix-vector applications are $\mathbf{Mv}$ and $\mathbf{v}^T\mathbf{M}$.

Teletraffic quantities

λ	arrival rate $[s^{-1}]$
μ	service rate $[s^{-1}]$
ρ	load $(= \lambda/\mu)$

Abbreviations

3GPP	Third Generation Partnership Project
LTE	Long Term Evolution
VoIP	Voice over IP
UMTS	Universal Mobile Telecommunications System
GPRS	General Packet Radio Service
EDGE	Enhanced Data Rates for GSM Evolution
UE	User Equipment
E-UTRAN	Evolved UMTS Terrestrial Radio Access Network
QPSK	Quadrature Phase Shift Keying
QAM	Quadrature Amplitude Modulation
OFDMA	Orthogonal Frequency-Division Multiple Access
SC-FDMA	Single-Carrier Frequency-Division Multiple Access
MIMO	Multiple Input Multiple Output
FDD	Frequency-Division Duplexing
TDD	Time-Division Duplexing
MDP	Markov Decision Process
SINR	Signal to Interference and Noise Ratio
SIR	Signal to Interference Ratio
AWGN	Additive White Gaussian Noise

1 Introduction

The generalisation of mobile Internet, and the increasing demand in bandwidth-hungry mobile services—such as real-time video, gaming, voice over IP, etc.—led wireless technologies to adopt packet-switched networks as their main component. First the General Packet Radio Service (GPRS) in 2G, and the Enhanced Data Rates for GSM Evolution (EDGE) in 3G allowed packet switched traffic in addition to the traditional circuit-switched networks. Long Term Evolution (LTE) systems are now based on all-IP core networks, and thus set a clear standard for the future of wireless cellular broadband access.

Elastic data traffic From an all-IP network's point of view, all concurrent data requests are streams of bytes or flows, seamlessly interwoven into one stream of packets. Since no circuits, or connections, are reserved for anyone, the actual data rate experienced by a particular user depends largely on

1. the number of other flows sharing the same resource, typically the number of active users in the same cell,
2. the quality of the radio channel for the user, which itself depends directly on the experienced signal to interference ratio. This makes inter-cell interference a major limitation on the data rates.

Since item 1 varies randomly, the data rates are variable, or "elastic". Although we can consider incoming users as an immutable given, item 2 on the other hand is related to the network and is therefore more likely to be controlled.

Coordination of base stations In [1], the authors introduce the new notion of reducing inter-cell interference by coordinating base stations according to the random nature of traffic. The basic motivation is that by turning off a base station, higher rates can be achieved in neighboring cells.

They introduce the notion of "capacity" of a cell from a traffic point of view (as opposed to static situations), and model a cellular network with many classes of users, depending on their location in the cell (and therefore their data rates). Assuming the base stations transmit at full power to one user at a time, they find the optimal static scheduling strategies, that is how to attribute regions to user classes depending on which base stations are on, in order to maximize data rates. Numerical applications are done in symmetric 2 and 3 cells networks. However, the optimization also involves load balancing which can be a contradictory objective, especially when interference is high.

Queuing model In [2], the authors simplify this model by removing the spatial dependency, and considering only two cells. This reduces to 2 classes of users (one per cell), with heterogeneous service requirement distributions. These two classes can be served either in parallel, or one after the other, which leads to different data rates, representing the presence or absence of interference. Since the data rates are now fixed, the objective is different, and more representative of a queuing model: minimizing the total number of users in the system (and therefore the average delay) by applying a time scheduling policy (switching base stations on and off at different points of time).

The authors conclude that an optimal policy can be found only in particular cases in this stochastic model, but not in general. They propose also fluid process models and asymptotically fluid optimal policies characterized by switching curves.

In this thesis, inspired by the stochastic model in [2], we simplify it further by considering symmetrical service requirements only. Although the paper contains the general result, no detailed proof is given. We derive here a complete proof with comments on the results and numerical examples.

This direct proof is possible only because of the drastic simplifications we assumed. In more general cases, a numerical method such as policy improvement—as described by the theory of Markov decision processes—may be needed. In order to demonstrate its efficiency, we apply it to the problem at stake, and show that it gives the same result.

Frequency reuse Another interference mitigation technique is studied in [3]: frequency reuse patterns. These are motivated by the fact that LTE's downlink is based on OFDMA, which allows the division of the bandwidth to simultaneously transmit to different classes of users. Data rates are directly proportional to bandwidth, so using less bandwidth in each cell might seem like a loss. However, by cleverly allocating bands to particular geographic regions, interfering base stations can be put at greater distances from each other, therefore largely reducing interference and in fact, increasing data rates.

Although reuse patterns were largely studied before in static situations, this paper uses the traffic-wise capacity as defined in [1] as a measure of their performance. Without deriving their analytical results, we re-compute the numerical results as a proof of concept, possibly to be later integrated with time scheduling for even better performance.

Outline This thesis is organized as follows. In Section 2, a brief description of the technologies related to the problem is given, while in Section 3 we introduce the mathematical tools that we use.

The main problem—finding an optimal way of realizing time scheduling between two base stations—is established and solved in Section 4. In Section 5 we solve the problem again but using Markov decision processes. Sections 6 and 7 present a numerical quantification of the gain brought by the optimal policy, as well as a discussion on its fairness.

The rather independent computation of capacity gains brought by frequency reuse patterns is done in Section 8.

Finally, Section 9 is a summary of the results, and considerations on future works to base on this one.

2 Technological background: 3GPP LTE

Long Term Evolution is the latest standard in mobile networking technology, developed by the 3rd Generation Partnership Project. Its goal is to answer to the increasing demand for mobile broadband services, as users are now used to have broadband access on personal computers and expect a similar quality of service regardless of the location and the device. Indeed, by 2014, it is estimated that about 3.4 billion people will have access to broadband, among which 80% through mobile subscriptions (see [4]).

LTE is a set of improvements over the Universal Mobile Telecommunications System (UMTS). Although it focuses on adopting 4G requirements, it does not fully comply with them, and in this sense is a pre-4G step towards LTE Advanced—an actual 4G compliant standard—, which is being developed by 3GPP as well.

Although the Internet is already accessible on 3G phones, LTE proposes to improve user experience by increasing data rates and reducing latency, thus enhancing applications such as web browsing, video streaming, voice over IP (VoIP), multimedia on-line gaming, real-time video, etc.

Performance goals In order to accommodate these highly demanding services in terms of data rates and latency, the LTE standard imposes the following performance requirements (retrieved from [5]):

Metric	Requirement
Peak data rate for 20 MHz spectrum	DL: 100 Mbps UL: 50 Mbps
Mobility support	Up to 500 km/h Optimized for 0 to 15 km/h
Control plane latency (time from idle to active state)	< 100 ms
User plane latency	< 5 ms
Control plane capacity for 5 MHz spectrum	> 200 users per cell
Coverage	5–100 km with slight degradation after 30 km
Spectrum flexibility	1.4, 3, 5, 10, 15 and 20 MHz

Additional goals are also to minimize power consumption and simplify user equipment as much as possible. Furthermore, handover from LTE to 2G/3G is designed to be seamless.

2.1 Evolved UMTS Terrestrial Radio Access Network

LTE's air interface is called E-UTRAN, for Evolved UMTS Terrestrial Radio Access Network. The Radio Access Network for LTE consists of a unique node that interfaces with user equipments, called eNodeB. A single eNodeB provides similar services than a nodeB and a Radio Network Controller together provided in UMTS, and therefore reduces latency.

On the physical layer, E-UTRAN uses Orthogonal Frequency Division Multiplexing (OFDM) for the downlink. The radio resource is divided into a time-frequency grid. In frequency domain, the bandwidth is divided into many narrow sub-carriers of width $\Delta f = 15$ kHz, while in time domain, the symbol duration is $1/\Delta f$ plus a cyclic prefix (used to maintain orthogonality between sub-carriers in time-dispersive radio channels). Each one of these resource elements carries a variable number of bits, depending on the modulation used (QPSK: 2 bits, 16-QAM: 4 bits, or 64-QAM: 6 bits). They are finally grouped into resource blocks of 12 sub-carriers and 7 symbols, that is, 180 kHz and time slots of about 5 ms.

The allocation of resource blocks to users is done according to advanced scheduling mechanisms, that can be applied as often as every millisecond.

In uplink, limiting power consumption is a major objective, since the signal is emitted from user equipments. However, OFDM has a high Peak to Average Power Ratio, which requires high quality power amplifiers (which would increase the cost of the equipment), and empties the battery faster, which is always a problem on mobile devices. To compensate this, in the uplink, LTE uses Single Carrier Frequency Division Multiple Access (SC-FDMA), a linearly precoded OFDM which effectively reduces power consumption and the need for efficient power amplifiers.

LTE also specifies multi-layered antenna technologies like 2x2 and 4x4 Multiple Input Multiple Output (MIMO), and beam-forming for extended coverage. Furthermore, both Frequency Division Duplexing (uplink and downling occuring at the same time on different frequency bands) and Time Division Multiplexing (uplink and downling alternating on the same frequency band) are available.

2.2 System Architecture Evolution

The core network architecture of LTE is called System Architecture Evolution (SAE), designed to improve and replace the General Packet Radio Service (GPRS) core network, by simplifying it. Its main component is the Evolved Packet Core (EPC): an all-IP network that supports higher throughput and lower latency RANs (Radio Access Networks, such as E-UTRAN, see Section 2.1), and mobility between previous RANs, GPRS for instance.

- The Serving Gateway routes and forwards packets, and manages inter eNodeB handovers, as well as mobility between LTE and other 3GPP technologies (2G/3G systems).
- The Mobility Management Entity is responsible for tracking idle user equipments, choosing a Serving Gateway for user equipments when they turn on. It also deals with intra-LTE handovers that require Core Network node relocation. Indeed, LTE enables sharing the eNodeBs between several Core

Networks (Mobility Management Entity, Serving Gateway and Packet Data Network Gateway). This allows service providers to reduce their costs by operating only a Core Network, and using the same eNodeBs conjointly.

- Finally, the Packet Data Network Gateway connects Serving Gateways to the external packet data networks (such as the Internet), and provides mobility to non 3GPP technologies (WiMAX, CDMA 1x, EvDO).

See [5] for a detailed description of SAE.

2.3 Simplified models

In this thesis, although we place ourselves in the very general context of LTE, the models we use are extremely simplified. In Sections 4 and 5, the two interfering base station can roughly model eNodeBs, but all physical or technical parameters are reduced to a simple queuing model determined by average throughputs.

In Section 8, we consider the division and allocation of bandwidth to different classes of users, which is inherent to a multiple access scheme such as OFDMA.

Beyond this, it should be noted that this thesis is of very theoretical nature, and not necessarily tied to any actual technology.

3 Theoretical background

Note: the theory of dynamic programming (introduced by Bellman in [6]) and Markov processes (see e.g. [7]) is wider than the following short introduction. We focus here on the elements necessary to the comprehension of the problem at stake.

3.1 Markov processes

Definition Let $X(t)$ be a discrete-state, continuous-time stochastic process. It is said to be a Markov process if

$$P[X(t_{n+1}) = x_{n+1} | X(t_1) = x_1, \dots, X(t_n) = x_n] = \\ P[X(t_{n+1}) = x_{n+1} | X(t_n) = x_n]$$

for all $n \in \mathbb{N}$, ordered time instants $t_1 < \dots < t_{n+1}$, and states $x_1, \dots, x_{n+1}$. In other words, given the current state, the future of the process does not depend on its past. We will also assume that $X(t)$ is time-homogeneous, that is,

$$P[X(t+h) = y | X(t) = x] = P[X(h) = y | X(0) = x]$$

for all positive times t, h and states x, y.

Transition rates Time-homogeneity allows us to define state transition rates q_{ij} that are themselves independent of the time instant t:

$$q_{ij} = \lim_{h \to 0} \frac{1}{h} P[X(h) = j | X(0) = i].$$

Equivalently, during a short time interval h, the probability of transition from state i to state j is

$$P[X(t+h) = j | X(t) = i] = q_{ij}h + o(h).$$

Therefore, the probability of transition from i to any other state during a short time interval h is

$$q_i h + o(h), \quad \text{with } q_i = \sum_{j \neq i} q_{ij}.$$

The memoryless assumption due to the Markov property makes this independent of any other time interval. This is of course the characteristics of an exponential distribution: the holding time in state i is exponentially distributed with rate q_i:

$$T_i \sim \mathrm{Exp}(q_i).$$

However, this holding time does not inform about which state the transition is done to. Let T_{ij} be the "potential" holding time for the $i \to j$ transition, that is, the holding time that there would be if $i \to j$ was the only possible transition. Again, we have

$$T_{ij} \sim \mathrm{Exp}(q_{ij}).$$

We can then express the holding time by

$$T_i = \min_{j \neq i} T_{ij},$$

and finally

$$P[T_i = T_{ij}] = \frac{q_{ij}}{q_i},$$

as proved in Appendix B.

State probabilities We can now find the state probabilities, that is the probability that the system is in a certain state at a given time

$$\pi_j(t) = P[X(t) = j].$$

For an infinitesimal time h, the probability of being in state j at time $t+h$ is the probability of having been in state i at time t and having had a $i \to j$ transition, plus the probability of being already in state j at time t and having had no transition:

$$\begin{aligned}
\pi_j(t+h) &= \sum_{i \neq j} \pi_i(t) q_{ij} h + \pi_j(t)(1 - q_j h) + o(h) \\
\pi_j(t+h) - \pi_j(t) &= \sum_{i \neq j} \pi_i(t) q_{ij} h - \pi_j(t) q_j h + o(h).
\end{aligned}$$

To facilitate vector notation, we take the convention $q_{jj} = -q_j$, which results in

$$\begin{aligned}
\pi_j(t+h) - \pi_j(t) &= h \sum_i \pi_i(t) q_{ij} + o(h) \\
\frac{\pi_j(t+h) - \pi_j(t)}{h} &= \sum_i \pi_i(t) q_{ij} + \frac{o(h)}{h} \\
\frac{\mathrm{d}\pi_j}{\mathrm{d}t}(t) &= \sum_i \pi_i(t) q_{ij},
\end{aligned}$$

or in vector form,

$$\frac{\mathrm{d}\boldsymbol{\pi}^T}{\mathrm{d}t}(t) = \boldsymbol{\pi}^T(t)\mathbf{Q}.$$

The general solution to this differential equation is given by

$$\boldsymbol{\pi}^T(t) = \boldsymbol{\pi}^T(0) e^{\mathbf{Q}t},$$

but the real interest is in the equilibrium distribution, which—if taken as initial distribution—makes the Markov process stationary:

$$\frac{\mathrm{d}\boldsymbol{\pi}^T}{\mathrm{d}t}(t) = \mathbf{0} \iff \boldsymbol{\pi}^T(t)\mathbf{Q} = \mathbf{0}.$$

We therefore call steady state distribution the distribution $\boldsymbol{\pi}$ that satisfies

$$\boldsymbol{\pi}^T \mathbf{Q} = \mathbf{0}.$$

For each column j, this equation expands to

$$\sum_i \pi_i q_{ij} = 0$$

$$\sum_{i \neq j} \pi_i q_{ij} + \pi_j q_{jj} = 0$$

$$\sum_{i \neq j} \pi_i q_{ij} - \pi_j \sum_{i \neq j} q_{ji} = 0,$$

and then

$$\sum_{i \neq j} \pi_i q_{ij} = \sum_{i \neq j} \pi_j q_{ji},$$

which is called the global balance equation, expressed for state j. Of course, to be a proper distribution, it should be normalized:

$$\sum_i \pi_i = 1 \iff \boldsymbol{\pi}^T \mathbf{e} = 1,$$

where $e_i = 1, \forall i$. It is also possible to express both conditions in a single equation:

$$\boldsymbol{\pi} = (\mathbf{Q}^T + \mathbf{E})^{-1} \mathbf{e},$$

where $\mathbf{E} = \mathbf{e}\mathbf{e}^T$ is the matrix with all elements equal to 1.

3.2 Markov decision processes

While continuous Markov processes are a useful tool in the study of stochastic systems, they have the significant disadvantage of describing only fixed situations, in which the transition rates are constant. It is sometimes necessary to let these rates vary, for instance to model the possibility of choice, or *decision*.

In a Markov decision process, to each state we associate a number of possible actions, which determine both the transition rates out of the state, and a revenue rate (or equivalently, a cost rate). A mapping of the states to the actions is called a policy, and naturally the aim of the theory is to find an optimal policy, which maximizes the average revenue (or minimizes the cost).

Although the concept can be applied to discrete time, we will focus here on continuous time, as the processes we will study later are continuous.

To comply with the Markov property, in each state i, the choice of an action a can depend only on the current state. We will denote the corresponding revenue rate $r_i(a)$ and the outgoing transition rates $q_{ij}(a)$. Since a policy α maps each state to a single action, we will in fact write these rates as functions of the policy: $r_i(\alpha)$, $q_{ij}(\alpha)$, or in vector form, $\mathbf{r}(\alpha)$ and $\mathbf{Q}(\alpha)$.

Note that fixing the policy α turns a Markov decision process into a simple Markov process. We can therefore solve from its transition rate matrix $\mathbf{Q}(\alpha)$ the equilibrium distribution $\boldsymbol{\pi}(\alpha)$:

$$\begin{cases} \boldsymbol{\pi}^T(\alpha)\mathbf{Q}(\alpha) = \mathbf{0} \\ \boldsymbol{\pi}^T(\alpha)\mathbf{e} = 1 \end{cases}.$$

Average revenue rate Once the equilibrium distribution $\boldsymbol{\pi}(\alpha)$ is solved, we get directly the average revenue rate $r(\alpha)$:

$$r(\alpha) = \sum_i \pi_i(\alpha) r_i(\alpha) = \boldsymbol{\pi}^T(\alpha)\mathbf{r}(\alpha).$$

The objective is now to find the optimal policy, that is the policy α^* that maximizes the average revenue (or minimizes the average cost, defined similarly):

$$\alpha^* = \arg\max_{\alpha} r(\alpha).$$

With a finite number of states and actions, there is a finite number of policies as well, which in principle, allows to compute the average revenue rate for all of them. In practice, this can be computationally impossible, and in many cases, the state space is infinite anyway.

Solving for the optimal policy is rather done using dynamic programming techniques, one of which being called policy iteration.

Note: from now on, the α parameter can be omitted for clarity of the notations, but it should be assumed that every value depends on the current policy.

Relative state values We know that, as time grows to infinity, the state probability vector gets closer and closer to the equilibrium distribution, regardless of its initial value:

$$\lim_{t\to\infty} \boldsymbol{\pi}^T(t) = \lim_{t\to\infty} \boldsymbol{\pi}^T(0) e^{\mathbf{Q}t} = \boldsymbol{\pi}^T.$$

However, different initial values may lead to a different short term evolution of the state probabilities. We introduce $V_i(t)$, the cumulative revenue starting from state i, that is the integral of the revenue rate over the time interval $[0, t]$.

By choosing $\boldsymbol{\pi}(0) = \mathbf{e}_i$ (the vector whose i^{th} component is 1 and the others zero) we have:

$$\begin{aligned} V_i(t) &= \int_0^t \boldsymbol{\pi}^T(u)\mathbf{r}\,\mathrm{d}u \\ &= \int_0^t \mathbf{e}_i^T e^{\mathbf{Q}u}\mathbf{r}\,\mathrm{d}u \\ &= \mathbf{e}_i^T \left(\int_0^t e^{\mathbf{Q}u}\,\mathrm{d}u \right) \mathbf{r}, \end{aligned}$$

and in vector form

$$\mathbf{V}(t) = \int_0^t e^{\mathbf{Q}u}\,\mathrm{d}u \cdot \mathbf{r}.$$

Since when t tends to infinity, the revenue rate tends to the average revenue rate, which is positive, the integral grows generally unbounded. We therefore use instead the *relative* state values v_i to compare the initial transient behaviors, and ignore the long term effects which are independent of the initial state:

$$\begin{aligned} v_i &= \int_0^\infty \boldsymbol{\pi}^T(u)\mathbf{r} - \boldsymbol{\pi}^T\mathbf{r}\,\mathrm{d}u \\ &= \lim_{t\to\infty} V_i(t) - t\cdot r \\ \mathbf{v} &= \lim_{t\to\infty} \mathbf{V}(t) - t\cdot r\mathbf{e}. \end{aligned}$$

Howard equation Because of the memorylessness of a Markov process, it is possible to find a relation between the relative state values. Indeed, *arriving* in a new state or *starting* in this same state should be strictly equivalent. Therefore we can separate the integral in two parts: before the first transition (which on average occurs at $E[T_i]$), and after:

$$\begin{aligned} v_i &= \int_0^\infty \boldsymbol{\pi}^T(u)\mathbf{r} - \boldsymbol{\pi}^T\mathbf{r}\,\mathrm{d}u \\ &= \int_0^{E[T_i]} \boldsymbol{\pi}^T(u)\mathbf{r} - \boldsymbol{\pi}^T\mathbf{r}\,\mathrm{d}u + \int_{E[T_i]}^\infty \boldsymbol{\pi}^T(u)\mathbf{r} - \boldsymbol{\pi}^T\mathbf{r}\,\mathrm{d}u. \end{aligned}$$

Between 0 and $E[T_i]$, the state probability collapses to $\mathbf{e}_i$, since we know with certainty to be in state i. After $E[T_i]$, we are in state j with probability q_{ij}/q_i, and the second integral is equal to v_j:

$$\begin{aligned} v_i &= \int_0^{E[T_i]} r_i - r\,\mathrm{d}u + \sum_{j\neq i} \frac{q_{ij}}{q_i} v_j \\ v_i &= \frac{1}{q_i}(r_i - r) + \sum_{j\neq i} \frac{q_{ij}}{q_i} v_j \\ q_i v_i &= r_i - r + \sum_{j\neq i} q_{ij} v_j. \end{aligned}$$

And since $q_i = -q_{ii}$, we finally get what is known as the Howard equation (introduced e.g. in [8]):

$$r_i(\alpha) - r(\alpha) + \sum_j q_{ij}(\alpha) v_j(\alpha) = 0, \quad \forall i,$$

or in vector form:

$$\mathbf{r}(\alpha) - r(\alpha)\mathbf{e} + \mathbf{Q}(\alpha)\mathbf{v}(\alpha) = \mathbf{0}.$$

Computational considerations So far, we started with the computation of the equilibrium distribution, then from it came the average revenue rate, which is used in the definition of the relative state values. The definition of the latter being somewhat difficult to use, we establish that they satisfy a certain relation (Howard equation), which is the practical way to compute them.

However, this implies solving two systems of equations: one for the equilibrium distribution, and one for the relative state values. If the ultimate goal is the relative state values, regardless of any other intermediate variables, it is in fact possible to solve only one system of equations.

Indeed, consider the following equation:

$$\mathbf{r}(\alpha) - g\mathbf{e} + \mathbf{Q}(\alpha)\mathbf{v}(\alpha) = \mathbf{0},$$

where g is unknown. Since each row of $\mathbf{Q}(\alpha)$ sums up to 0, $\mathbf{Qe} = \mathbf{0}$. Consequently, if $\mathbf{v}$ is a solution to this equation, $\mathbf{v} + c \cdot \mathbf{e}$ is as well. In other words, the $v_i(\alpha)$ are determined up to an additive constant, and we can set for instance $v_1(\alpha) = 0$. This allows to solve g as an unknown in the problem, which keeps an equal number of equations and variables.

Now if g and $\mathbf{v}$ are solution to this equation, multiplying it by $\boldsymbol{\pi}^T$ gives:

$$\begin{aligned}\boldsymbol{\pi}^T\mathbf{r} - g\boldsymbol{\pi}^T\mathbf{e} + \boldsymbol{\pi}^T\mathbf{Qv} &= \boldsymbol{\pi}^T\mathbf{0}\\ r - g \cdot 1 + \mathbf{0v} &= \mathbf{0},\end{aligned}$$

and finally,

$$g = r(\alpha).$$

In conclusion, it is possible to solve for the average revenue rate and the relative state values without knowing the steady state distribution.

Policy iteration From a policy α, we build a new policy α' by choosing in each state i the action a_i according to the following definition:

$$a_i = \arg\max_a \left\{ r_i(a) - r(\alpha) + \sum_j q_{ij}(a)v_j(\alpha) \right\}. \tag{3.1}$$

Intuitively, this corresponds to taking the action that maximizes the immediate revenue rate, and the expected revenue for the rest of the time (if after the next transition the old policy was used).

It is shown in [9] that the obtained policy is never worse that the original. That is,

$$r(\alpha') \geq r(\alpha).$$

Once the new policy is found, the new relative state values are computed, and the policy iteration can be applied again until convergence. Indeed, if the policy does not change in the iteration process, then it is proved that it is in fact optimal.

Relative state values in an M/M/1 queue As an example—and because the result will be needed later—we derive here the relative state values in an M/M/1 queue, with arrival rate λ and service rate μ.

Note that we do not mention any policy in what follows. Since we consider the "usual" M/M/1 queue, which is a simple Markov process, we assume a free-access policy. The only available action is to serve at the constant rate μ, and accept all incoming users. The transition rates and costs are constant.

The Markov process is as follows:

- the states $n \in \mathbb{N}$ represent the queue length
- the transition rates are between neighboring states only (which makes the queue a birth-death process)

$$q_{n,n+1} = \lambda, \quad \forall n \in \mathbb{N}$$
$$q_{n,n-1} = \mu, \quad \forall n > 0.$$

Since the ultimate goal will be to minimize the delay in the system, we will consider costs instead of revenues. By Little's formula, the average delay in a stable system is proportional to the average number of users present in the system (proved in [10]). It is therefore reasonable to use the number of users (i.e. the length of the queue) as the immediate cost rate.

$$r_n = n.$$

The Howard equation follows directly:

$$\begin{aligned} v_0 &= 0 \\ -r + \lambda v_1 - \lambda v_0 &= 0 \\ n - r + \lambda v_{n+1} + \mu v_{n-1} - (\lambda + \mu) v_n &= 0, \quad \forall n > 0. \end{aligned}$$

It is easy to check that

$$v_n = \frac{1}{2} \frac{n(n+1)}{\mu - \lambda}, \quad r = \frac{\lambda}{\mu - \lambda} \tag{3.2}$$

verifies the equation and is therefore the solution. Note that since we chose the queue length for the cost rate r_n, r represents the average queue length, which is indeed known to be

$$E[N] = \frac{\lambda}{\mu - \lambda} \tag{3.3}$$

for the M/M/1 queue[1].

[1] It can be otherwise computed by finding the equilibrium state distribution and computing its average, since the states n have been taken to represent queue lengths.

4 Time scheduling of two interfering base stations

Two base stations using the same frequency band interfere, in that a user in one cell will hear the signal from the other cell as unwanted noise which competes with the signal coming from its own base station. In other words, a neighboring base station decreases its own signal to interference ratio, thus reducing the data rate he experiences.

From the simple remark that a base station does not interfere when it is turned off, we can intuitively infer that it should be possible to increase the global performance of a network by turning off the base stations when they are least needed.

The concept of time scheduling for base stations—according to incoming random traffic—has been introduced in [1], where the exact locations of the users in the cells were considered. It was then simplified in [2], where the spatial component was removed by averaging rates over the cell's area.

We consider here two interfering base stations (see Figure 1), each with its own queue of arriving customers, starting downlink flows of random sizes. We make the simplifying assumption that the flow sizes are identically distributed, as it should be if all users use the same kinds of services. However, we allow arrival rates in the two cells to be different, since the two cells could be for instance of different sizes, or in areas populated differently.

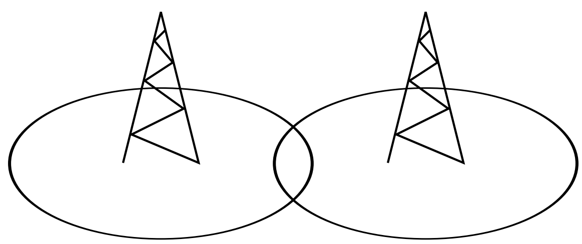

Figure 1: Interfering base stations

By "time scheduling" we mean that at any time, each base station can be either on or off. We give ourselves the objective of minimizing the average total number of users in the system. Since this is directly related to the average delay experienced, this optimization is positive in terms of quality of service.

4.1 Queuing model description

We model the situation as follows. Consider two classes of users ($i = 1, 2$) arriving according to independent Poisson processes with rate λ_i, and having the same exponentially distributed service requirements with mean $E[X]$ (bits). At any time, the server has three possible actions: serve one class only (either class 1 or class 2) with service rate c_0, or serve both classes 1 and 2 in parallel with rate c_1 for each (bits/s). To model the fact that the stations interfere with each other when serving at the same time, we assume $c_1 < c_0$. As we will see, the relative values of c_1 and

c_0 have a major impact. The case $2c_1 > c_0$ will be referred to as "low interference", and its counterpart $2c_1 \leq c_0$ as "high interference".

We denote by μ_i the departure rates associated with rates c_i, that is:

$$\mu_0 = \frac{c_0}{E[X]}, \quad \mu_1 = \frac{c_1}{E[X]}.$$

For a given policy π, we denote by $N_i^\pi(t)$ the number of class-i users in the system at time t, $N^\pi(t) = (N_1^\pi(t), N_2^\pi(t))$. If the random variables with steady-state distributions exist, they are denoted as N_i^π and N^π. By considering only non-anticipating policies Π, the exponential service requirements make the stochastic behavior independent of the discipline applied within a class. We can assume for instance Processor Sharing.

A policy $\tilde{\pi}$ is called average optimal if

$$\tilde{\pi} = \text{argmin}_{\pi \in \Pi} E[N_1^\pi + N_2^\pi],$$

and stochastically optimal if

$$N_1^{\tilde{\pi}}(t) + N_2^{\tilde{\pi}}(t) \leq_{st} N_1^\pi(t) + N_2^\pi(t), \quad \forall t \geq 0, \forall \pi \in \Pi, \tag{4.1}$$

where $X \leq_{st} Y$ means $P[X > s] \leq P[Y > s], \quad \forall s \geq 0$. Note that stochastic optimality implies average optimality.

4.2 Stochastically optimal policy in low interference

Here we prove that when $2c_1 > c_0$, the stochastically optimal policy is to have both stations switched on whenever possible.

Let the functions $V_k, k \in \mathbb{N}$ be defined as follows:

$$\begin{aligned} V_0(x) &= \mathbf{1}_{(x_1+x_2>s)} \\ V_{k+1}(x) &= \frac{\lambda_1}{\nu} V_k(x_1+1, x_2) + \frac{\lambda_2}{\nu} V_k(x_1, x_2+1) \\ &\quad + \min\Big\{ \frac{\mu_0}{\nu} V_k((x_1-1)^+, x_2) + \frac{2\mu_1}{\nu} V_k(x), \\ &\qquad \frac{\mu_0}{\nu} V_k(x_1, (x_2-1)^+) + \frac{2\mu_1}{\nu} V_k(x), \\ &\qquad \frac{\mu_1}{\nu} V_k((x_1-1)^+, x_2) + \frac{\mu_1}{\nu} V_k(x_1, (x_2-1)^+) + \frac{\mu_0}{\nu} V_k(x) \Big\}, \end{aligned}$$

for $x_1, x_2 \geq 0$, $\nu = \lambda_1 + \lambda_2 + \mu_0 + 2\mu_1$ and $s > 0$. Since we are interested in the minimization problem, the $\frac{1}{\nu}$ factor—present in every term—can be omitted without loss of generality.

The $V_k(x)$ function represents the minimal expected cost after k transitions from a (x_1, x_2) initial state, in the uniformized Markov process[2]. Each term in the recursive definition of V_{k+1} is the probability of transitioning from some state at step k

[2] Since the transitions depend on the choice of an action, we need to add dummy transitions to the same state, so that whatever the action chosen, the sum of transition rates is the same, denoted by ν.

multiplied by the expected cost of this state at the previous step. Each option in the min{} represents one action (station 1 on, station 2 on, both stations on), and uses transition probabilities which depend on the associated achieved rates.

Proposition 1. *$V_k(x)$ is non decreasing in x_1 and x_2 for all k.*

Proof. By induction. Assuming V_k is non decreasing in x_1, x_2, all terms of V_{k+1} are of the form $cV_k(f_d(x_1), f_e(x_2))$ where $c > 0$ and $f_d(x_i) = (x_i + d)^+$, and are thus non decreasing functions of x_i.

$V_0 = \mathbf{1}_{(x_1+x_2>s)}$ is obviously non decreasing in x_1 and x_2. □

Proposition 2. *For $x_1 > 0, x_2 = 0$, $k \in \mathbb{N}$,*

$$\begin{aligned}\mu_0 V_k(x_1-1,x_2) + 2\mu_1 V_k(x)&\\ \le \min\Big\{&\mu_0 V_k(x_1,(x_2-1)^+) + 2\mu_1 V_k(x),\\ &\mu_0 V_k(x) + \mu_1 V_k(x_1-1,x_2) + \mu_1 V_k(x_1,(x_2-1)^+)\Big\},\end{aligned}$$

(and inversely for $x_1 = 0, x_2 > 0$). In other words, the minimizing term in V_k is the one associated with the action of serving only the non empty queue.

Proof. With $x_2 = 0$, the first inequality to prove reduces to

$$\mu_0 V_k(x_1-1,0) + 2\mu_1 V_k(x) \le \mu_0 V_k(x_1,0) + 2\mu_1 V_k(x),$$

and finally

$$V_k(x_1-1,0) \le V_k(x_1,0),$$

which is true, by the previous proposition.

The second inequality reduces to

$$\begin{aligned}\mu_0 V_k(x_1-1,0) + 2\mu_1 V_k(x_1,0) &\le \mu_0 V_k(x) + \mu_1 V_k(x_1-1,0) + \mu_1 V_k(x_1,0)\\ \mu_0 V_k(x_1-1,0) + \mu_1 V_k(x_1,0) &\le \mu_0 V_k(x) + \mu_1 V_k(x_1-1,0)\\ \mu_1[V_k(x_1,0) - V_k(x_1-1,0)] &\le \mu_0[V_k(x_1,0) - V_k(x_1-1,0)],\end{aligned}$$

which is always true, since by previous proposition $V_k(x_1,0) - V_k(x_1-1,0) \ge 0$, and we assumed $c_1 < c_0$ ($\iff \mu_1 < \mu_0$).

The proof is similar with $x_1 = 0, x_2 > 0$. □

Proposition 3. *For $x_1, x_2 > 0$, $k \in \mathbb{N}$,*

$$\begin{aligned}\mu_0 V_k(x)+&\mu_1 V_k(x_1-1,x_2) + \mu_1 V_k(x_1,x_2-1)\\ \le \min\Big\{&\mu_0 V_k(x_1,x_2-1) + 2\mu_1 V_k(x),\\ &\mu_0 V_k(x_1-1,x_2) + 2\mu_1 V_k(x)\Big\}.\end{aligned}$$

Proof. By induction.

Basis of induction At $k = 0$, $V_0 = \mathbf{1}_{(x_1+x_2>s)}$, so $V_0(x_1-1, x_2) = V_0(x_1, x_2-1)$, and the inequality to prove is reduced to:

$$\mu_0 \mathbf{1}_{(x_1+x_2>s)} + 2\mu_1 \mathbf{1}_{(x_1+x_2>s+1)} \leq \mu_0 \mathbf{1}_{(x_1+x_2>s+1)} + 2\mu_1 \mathbf{1}_{(x_1+x_2>s)},$$

which is equivalent to

$$c_0 f(x) \leq 2c_1 f(x),$$

where $f(x) = \mathbf{1}_{(x_1+x_2>s)} - \mathbf{1}_{(x_1+x_2>s+1)}$. Note that

$$f(x) = \begin{cases} 1 & \text{if } x_1 + x_2 = s+1 \\ 0 & \text{otherwise} \end{cases},$$

so $f(x) \geq 0$. And since $c_0 < 2c_1$, the inequality is always verified.

Inductive step We assume it holds for V_k, and prove this implies that it holds for V_{k+1} also.

Case $x_1, x_2 > 1$ Since the inequality holds for V_k, the minimizing term in the definition of V_{k+1} is always the third one, giving the following explicit expression:

$$\begin{aligned}
&\mu_0 V_{k+1}(x) + \mu_1 V_{k+1}(x_1-1, x_2) + \mu_1 V_{k+1}(x_1, x_2-1) \\
=&\mu_0 \left[\lambda_1 V_k(x_1+1, x_2) + \lambda_2 V_k(x_1, x_2+1) + \mu_1 V_k(x_1-1, x_2) + \mu_1 V_k(x_1, x_2-1) + \mu_0 V_k(x)\right] \\
+&\mu_1 \left[\lambda_1 V_k(x) + \lambda_2 V_k(x_1-1, x_2+1) + \mu_1 V_k(x_1-2, x_2) + \mu_1 V_k(x_1-1, x_2-1) + \mu_0 V_k(x_1-1, x_2)\right] \\
+&\mu_1 \left[\lambda_1 V_k(x_1+1, x_2-1) + \lambda_2 V_k(x) + \mu_1 V_k(x_1-1, x_2-1) + \mu_1 V_k(x_1, x_2-2) + \mu_0 V_k(x_1, x_2-1)\right].
\end{aligned}$$

Rearranging the terms gives:

$$\begin{aligned}
&= \lambda_1[\mu_0 V_k(x_1+1, x_2) + \mu_1 V_k(x) + \mu_1 V_k(x_1+1, x_2-1)] \\
&+ \lambda_2[\mu_0 V_k(x_1, x_2+1) + \mu_1 V_k(x_1-1, x_2+1) + \mu_1 V_k(x)] \\
&+ \mu_1[\mu_0 V_k(x_1-1, x_2) + \mu_1 V_k(x_1-2, x_2) + \mu_1 V_k(x_1-1, x_2-1)] \\
&+ \mu_1[\mu_0 V_k(x_1, x_2-1) + \mu_1 V_k(x_1-1, x_2-1) + \mu_1 V_k(x_1, x_2-2)] \\
&+ \mu_0[\mu_0 V_k(x) + \mu_1 V_k(x_1-1, x_2) + \mu_1 V_k(x_1, x_2-1)].
\end{aligned}$$

At this point we recognize in each bracket the left side of the inequality assumed true for each V_k:

$$\begin{aligned}
&\leq \lambda_1[2\mu_1 V_k(x_1+1, x_2) + \mu_0 V_k(x_1+1, x_2-1)] \\
&+ \lambda_2[2\mu_1 V_k(x_1, x_2+1) + \mu_0 V_k(x)] \\
&+ \mu_1[2\mu_1 V_k(x_1-1, x_2) + \mu_0 V_k(x_1-1, x_2-1)] \\
&+ \mu_1[2\mu_1 V_k(x_1, x_2-1) + \mu_0 V_k(x_1, x_2-2)] \\
&+ \mu_0[2\mu_1 V_k(x) + \mu_0 V_k(x_1, x_2-1)],
\end{aligned}$$

and finally:

$$\begin{aligned}
&= \mu_0 \left[\lambda_1 V_k(x_1+1, x_2-1) + \lambda_2 V_k(x) + \mu_1 V_k(x_1-1, x_2-1) + \mu_1 V_k(x_1, x_2-2) + \mu_0 V_k(x_1, x_2-1)\right] \\
&+ 2\mu_1 \left[\lambda_1 V_k(x_1+1, x_2) + \lambda_2 V_k(x_1, x_2+1) + \mu_1 V_k(x_1-1, x_2) + \mu_1 V_k(x_1, x_2-1) + \mu_0 V_k(x)\right] \\
&= \mu_0 V_{k+1}(x_1, x_2-1) + 2\mu_1 V_{k+1}(x).
\end{aligned}$$

We have shown that for $x_1, x_2 > 1$,

$$\mu_0 V_{k+1}(x) + \mu_1 V_{k+1}(x_1-1, x_2) + \mu_1 V_{k+1}(x_1, x_2-1) \leq \mu_0 V_{k+1}(x_1, x_2-1) + 2\mu_1 V_{k+1}(x).$$

It can be shown in a similar way that

$$\mu_0 V_{k+1}(x) + \mu_1 V_{k+1}(x_1-1, x_2) + \mu_1 V_{k+1}(x_1, x_2-1) \leq \mu_0 V_{k+1}(x_1-1, x_2) + 2\mu_1 V_{k+1}(x).$$

Case $x_1 \geq 1, x_2 = 1$, first inequality Consider the following expression:

$$\mu_0 V_{k+1}(x_1, 1) + \mu_1 V_{k+1}(x_1-1, 1) + \mu_1 V_{k+1}(x_1, 0).$$

We can expand the first two terms by using the assumed property of V_k in its definition (and choosing the non-optimal policy that serves the second queue only, hence the inequality), and the third by application of Proposition 2:

$$\begin{aligned}
\leq &\mu_0[\lambda_1 V_k(x_1+1, 1) + \lambda_2 V_k(x_1, 2) + \mu_0 V_k(x_1, 0) + 2\mu_1 V_k(x_1, 1)] \\
+ &\mu_1[\lambda_1 V_k(x_1, 1) + \lambda_2 V_k(x_1-1, 2) + \mu_0 V_k(x_1-1, 0) + 2\mu_1 V_k(x_1-1, 1)] \\
+ &\mu_1[\lambda_1 V_k(x_1+1, 0) + \lambda_2 V_k(x_1, 1) + \mu_0 V_k(x_1-1, 0) + 2\mu_1 V_k(x_1, 0)].
\end{aligned}$$

Rearranging:

$$\begin{aligned}
= &\lambda_1[\mu_0 V_k(x_1+1, 1) + \mu_1 V_k(x_1, 1) + \mu_1 V_k(x_1+1, 0)] \\
+ &\lambda_2[\mu_0 V_k(x_1, 2) + \mu_1 V_k(x_1-1, 2) + \mu_1 V_k(x_1, 1)] \\
+ &\mu_0[\mu_0 V_k(x_1, 0) + 2\mu_1 V_k(x_1-1, 0)] \\
+ &2\mu_1[\mu_0 V_k(x_1, 1) + \mu_1 V_k(x_1-1, 1) + \mu_1 V_k(x_1, 0)].
\end{aligned}$$

In the first two terms we apply the assumed property of V_k. In the third we make use of the fact that

$$\begin{aligned}
&\mu_0 V_k(x_1, 0) + 2\mu_1 V_k(x_1-1, 0) \leq \mu_0 V_k(x_1-1, 0) + 2\mu_1 V_k(x_1, 0) \\
\iff &(\mu_0 - 2\mu_1)[V_k(x_1, 0) - V_k(x_1-1, 0)] \leq 0,
\end{aligned}$$

—which is true, since V_k is non decreasing by Proposition 1 and $\mu_0 < 2\mu_1$—and obtain:

$$\begin{aligned}
\leq &\lambda_1[\mu_0 V_k(x_1+1, 0) + 2\mu_1 V_k(x_1+1, 1)] \\
+ &\lambda_2[\mu_0 V_k(x_1, 1) + 2\mu_1 V_k(x_1, 2)] \\
+ &\mu_0[\mu_0 V_k(x_1-1, 0) + 2\mu_1 V_k(x_1, 0)] \\
+ &2\mu_1[\mu_0 V_k(x_1, 1) + \mu_1 V_k(x_1-1, 1) + \mu_1 V_k(x_1, 0)].
\end{aligned}$$

Finally:

$$\begin{aligned}
&= \mu_0[\lambda_1 V_k(x_1+1,0) + \lambda_2 V_k(x_1,1) + \mu_0 V_k(x_1-1,0) + 2\mu_1 V_k(x_1,0)] \\
&+ 2\mu_1[\lambda_1 V_k(x_1+1,1) + \lambda_2 V_k(x_1,2) + \mu_0 V_k(x_1,1) + \mu_1 V_k(x_1-1,1) + \mu_1 V_k(x_1,0)] \\
&= \mu_0 V_{k+1}(x_1,0) + 2\mu_1 V_{k+1}(x_1,1).
\end{aligned}$$

We have shown that for $x_1 \geq 1, x_2 = 1$,

$$\mu_0 V_{k+1}(x_1,1) + \mu_1 V_{k+1}(x_1-1,1) + \mu_1 V_{k+1}(x_1,0) \leq \mu_0 V_{k+1}(x_1,0) + 2\mu_1 V_{k+1}(x_1,1).$$

Case $x_1 > 1, x_2 = 1$, second inequality Again we consider the expression

$$\mu_0 V_{k+1}(x_1,1) + \mu_1 V_{k+1}(x_1-1,1) + \mu_1 V_{k+1}(x_1,0)$$

and expand it in a similar fashion. The first two terms are rewritten using the non-optimal policy that serves the first queue only (by the assumed property of V_k in its definition), and the third by application of Proposition 2:

$$\begin{aligned}
\leq & \mu_0[\lambda_1 V_k(x_1+1,1) + \lambda_2 V_k(x_1,2) + \mu_0 V_k(x_1-1,1) + 2\mu_1 V_k(x_1,1)] \\
+ & \mu_1[\lambda_1 V_k(x_1,1) + \lambda_2 V_k(x_1-1,2) + \mu_0 V_k(x_1-2,1) + 2\mu_1 V_k(x_1-1,1)] \\
+ & \mu_1[\lambda_1 V_k(x_1+1,0) + \lambda_2 V_k(x_1,1) + \mu_0 V_k(x_1-1,0) + 2\mu_1 V_k(x_1,0)]
\end{aligned}$$

$$\begin{aligned}
= & \lambda_1[\mu_0 V_k(x_1+1,1) + \mu_1 V_k(x_1,1) + \mu_1 V_k(x_1+1,0)] \\
+ & \lambda_2[\mu_0 V_k(x_1,2) + \mu_1 V_k(x_1-1,2) + \mu_1 V_k(x_1,1)] \\
+ & \mu_0[\mu_0 V_k(x_1-1,1) + \mu_1 V_k(x_1-2,1) + \mu_1 V_k(x_1-1,0)] \\
+ & 2\mu_1[\mu_0 V_k(x_1,1) + \mu_1 V_k(x_1-1,1) + \mu_1 V_k(x_1,0)].
\end{aligned}$$

In the first two terms we apply the assumed property of V_k and replace them by the expression associated with serving the first queue only. We leave the third and fourth terms untouched:

$$\begin{aligned}
\leq & \lambda_1[\mu_0 V_k(x_1,1) + 2\mu_1 V_k(x_1+1,1)] \\
+ & \lambda_2[\mu_0 V_k(x_1-1,2) + 2\mu_1 V_k(x_1,2)] \\
+ & \mu_0[\mu_0 V_k(x_1-1,1) + \mu_1 V_k(x_1-2,1) + \mu_1 V_k(x_1-1,0)] \\
+ & 2\mu_1[\mu_0 V_k(x_1,1) + \mu_1 V_k(x_1-1,1) + \mu_1 V_k(x_1,0)].
\end{aligned}$$

Finally:

$$\begin{aligned}
= & \mu_0[\lambda_1 V_k(x_1,1) + \lambda_2 V_k(x_1-1,2) + \mu_0 V_k(x_1-1,1) + \mu_1 V_k(x_1-2,1) + \mu_1 V_k(x_1-1,0)] \\
+ & 2\mu_1[\lambda_1 V_k(x_1+1,1) + \lambda_2 V_k(x_1,2) + \mu_0 V_k(x_1,1) + \mu_1 V_k(x_1-1,1) + \mu_1 V_k(x_1,0)] \\
= & \mu_0 V_{k+1}(x_1-1,1) + 2\mu_1 V_{k+1}(x_1,1).
\end{aligned}$$

We have shown that for $x_1 > 1, x_2 = 1$,

$$\mu_0 V_{k+1}(x_1,1) + \mu_1 V_{k+1}(x_1-1,1) + \mu_1 V_{k+1}(x_1,0) \leq \mu_0 V_{k+1}(x_1-1,1) + 2\mu_1 V_{k+1}(x_1,1),$$

thus completing the induction for the case $x_1 > 1, x_2 = 1$. The two inequalities of case $x_1 = 1, x_2 > 1$ are proven similarly.

Case $x_1 = 1, x_2 = 1$ We consider the expression

$$\mu_0 V_{k+1}(1,1) + \mu_1 V_{k+1}(0,1) + \mu_1 V_{k+1}(1,0)$$

and expand it using induction for the first term, and Proposition 2 for the second and third:

$$\begin{aligned}
=&\mu_0[\lambda_1 V_k(2,1) + \lambda_2 V_k(1,2) + \mu_0 V_k(0,1) + 2\mu_1 V_k(1,1)]\\
+&\mu_1[\lambda_1 V_k(1,1) + \lambda_2 V_k(0,2) + \mu_0 V_k(0,0) + 2\mu_1 V_k(0,1)]\\
+&\mu_1[\lambda_1 V_k(2,0) + \lambda_2 V_k(1,1) + \mu_0 V_k(0,0) + 2\mu_1 V_k(1,0)]\\
\\
=&\lambda_1[\mu_0 V_k(2,1) + \mu_1 V_k(1,1) + \mu_1 V_k(2,0)]\\
+&\lambda_2[\mu_0 V_k(1,2) + \mu_1 V_k(0,2) + \mu_1 V_k(1,1)]\\
+&\mu_0[\mu_0 V_k(0,1) + 2\mu_1 V_k(0,0)]\\
+&2\mu_1[\mu_0 V_k(1,1) + \mu_1 V_k(0,1) + \mu_1 V_k(1,0)].
\end{aligned}$$

In the first two terms we apply the assumed property of V_k. In the third term we again make use of the fact that V_k is non decreasing in x_1, x_2, and $\mu_0 < 2\mu_1$, which implies $\mu_0 V_k(0,1) + 2\mu_1 V_k(0,0) \leq \mu_0 V_k(0,0) + 2\mu_1 V_k(0,1)$. We leave the fourth term as it is:

$$\begin{aligned}
\leq&\lambda_1[\mu_0 V_k(1,1) + 2\mu_1 V_k(2,1)]\\
+&\lambda_2[\mu_0 V_k(0,2) + 2\mu_1 V_k(1,2)]\\
+&\mu_0[\mu_0 V_k(0,0) + 2\mu_1 V_k(0,1)]\\
+&2\mu_1[\mu_0 V_k(1,1) + \mu_1 V_k(0,1) + \mu_1 V_k(1,0)]\\
\\
=&\mu_0[\lambda_1 V_k(1,1) + \lambda_2 V_k(0,2) + \mu_0 V_k(0,0) + 2\mu_1 V_k(0,1)]\\
+&2\mu_1[\lambda_1 V_k(2,1) + \lambda_2 V_k(1,2) + \mu_0 V_k(1,1) + \mu_1 V_k(0,1) + \mu_1 V_k(1,0)]\\
\\
=&\mu_0 V_{k+1}(0,1) + 2\mu_1 V_{k+1}(1,1).
\end{aligned}$$

We have shown

$$\mu_0 V_{k+1}(1,1) + \mu_1 V_{k+1}(0,1) + \mu_1 V_{k+1}(1,0) \leq \mu_0 V_{k+1}(0,1) + 2\mu_1 V_{k+1}(1,1).$$

The second inequality

$$\mu_0 V_{k+1}(1,1) + \mu_1 V_{k+1}(0,1) + \mu_1 V_{k+1}(1,0) \leq \mu_0 V_{k+1}(1,0) + 2\mu_1 V_{k+1}(1,1)$$

was covered by the case $x_1 \geq 1, x_2 = 1$, thus completing the induction and the proof. □

In light of the previous propositions, we deduce that the minimizing action in the definition of the V_k is to use both servers in parallel when $x_1, x_2 > 0$ (that is when both queues are non empty), and to serve only the non empty queue when one of them is empty. In other words, turn off a base station if and only if it has no users to serve.

Finally, given the choice of the cost function $V_0(x) = \mathbf{1}_{(x_1+x_2>s)}$, the functions V_k can be interpreted as $V_k(x) = P[N_1(k) + N_2(k) > s | N(0) = x]$. Therefore, by showing that the minimizing action is always the same for all s and k, we have shown that the associated policy is stochastically optimal.

4.3 Stochastically optimal policy in high interference

It is possible to imagine cases where interference between the stations is so bad, that in the resulting model, $2c_1 \leq c_0$. Intuitively we can infer that using both stations at the same time is never in our interest, and that the optimal policy is to use them alternatively, one at a time.

Here we prove that it is indeed the case.

Propositions 1 and 2 still hold, since their proofs do not rely on the relative order of $2c_1$ and c_0.

Proposition 4. *For $x_1, x_2 > 0$, $k \in \mathbb{N}$,*

$$\begin{aligned} \mu_0 V_k(x) + &\mu_1 V_k(x_1 - 1, x_2) + \mu_1 V_k(x_1, x_2 - 1) \\ &\geq \max \Big\{ \mu_0 V_k(x_1, x_2 - 1) + 2\mu_1 V_k(x), \\ &\qquad \mu_0 V_k(x_1 - 1, x_2) + 2\mu_1 V_k(x) \Big\}. \end{aligned}$$

In other words, the minimizing term in the definition of V_k is never the one associated with serving both queues at the same time.

Proof. Similar to proof of Proposition 3, where all inequalities are inversed. □

Note that this proposition does not give information about the actual minimizing action.

Proposition 5. *In the definition of V_k, the terms associated with serving one queue only are equal. Formally, for $x_1, x_2 > 0$, $k \in \mathbb{N}$,*

$$\mu_0 V_k(x_1, x_2 - 1) + 2\mu_1 V_k(x) = \mu_0 V_k(x_1 - 1, x_2) + 2\mu_1 V_k(x),$$

or equivalently,

$$V_k(x_1, x_2 - 1) = V_k(x_1 - 1, x_2).$$

Proof. By induction.

At $k = 0$, $V_0 = \mathbf{1}_{(x_1+x_2>s)}$, so $V_0(x_1 - 1, x_2) = V_0(x_1, x_2 - 1)$, and the equality is immediate.

For $k > 0$, suppose that the equality holds for V_k. In light of the previous proposition, we can write, for $x_1, x_2 > 1$:

$$\begin{aligned} &\mu_0 V_{k+1}(x_1 - 1, x_2) + 2\mu_1 V_{k+1}(x) \\ =&\mu_0[\lambda_1 V_k(x) + \lambda_2 V_k(x_1 - 1, x_2 + 1) + \mu_0 V_k(x_1 - 2, x_2) + 2\mu_1 V_k(x_1 - 1, x_2)] \\ &+ 2\mu_1[\lambda_1 V_k(x_1 + 1, x_2) + \lambda_2 V_k(x_1, x_2 + 1) + \mu_0 V_k(x_1 - 1, x_2) + 2\mu_1 V_k(x)]. \end{aligned}$$

This can be rearranged to:

$$\begin{aligned}
=&\lambda_1[\mu_0 V_k(x) + 2\mu_1 V_k(x_1+1, x_2)]\\
+&\lambda_2[\mu_0 V_k(x_1-1, x_2+1) + 2\mu_1 V_k(x_1, x_2+1)]\\
+&\mu_0[\mu_0 V_k(x_1-2, x_2) + 2\mu_1 V_k(x_1-1, x_2)]\\
+&2\mu_1[\mu_0 V_k(x_1-1, x_2) + 2\mu_1 V_k(x)].
\end{aligned}$$

In each bracket we recognize the left side of the equality assumed true for V_k:

$$\begin{aligned}
=&\lambda_1[\mu_0 V_k(x_1+1, x_2-1) + 2\mu_1 V_k(x_1+1, x_2)]\\
+&\lambda_2[\mu_0 V_k(x_1, x_2) + 2\mu_1 V_k(x_1, x_2+1)]\\
+&\mu_0[\mu_0 V_k(x_1-1, x_2-1) + 2\mu_1 V_k(x_1-1, x_2)]\\
+&2\mu_1[\mu_0 V_k(x_1, x_2-1) + 2\mu_1 V_k(x)],
\end{aligned}$$

and finally:

$$\begin{aligned}
=&\mu_0[\lambda_1 V_k(x_1+1, x_2-1) + \lambda_2 V_k(x_1, x_2) + \mu_0 V_k(x_1-1, x_2-1) + 2\mu_1 V_k(x_1, x_2-1)]\\
+&2\mu_1[\lambda_1 V_k(x_1+1, x_2) + \lambda_2 V_k(x_1, x_2+1) + \mu_0 V_k(x_1-1, x_2) + 2\mu_1 V_k(x)]\\
=&\mu_0 V_{k+1}(x_1, x_2-1) + 2\mu_1 V_{k+1}(x).
\end{aligned}$$

We are left with the cases $x_1 = 1$ or $x_2 = 1$. Suppose $x_1 = 1, x_2 > 0$. Proposition 2 gives us the following expression for $V_{k+1}(0, x_2)$:

$$\lambda_1 V_k(1, x_2) + \lambda_2 V_k(0, x_2+1) + \mu_0 V_k(0, x_2-1) + 2\mu_1 V_k(0, x_2).$$

We therefore have:

$$\begin{aligned}
&\mu_0 V_{k+1}(0, x_2) + 2\mu_1 V_{k+1}(1, x_2)\\
=&\lambda_1[\mu_0 V_k(1, x_2) + 2\mu_1 V_k(2, x_2)]\\
+&\lambda_2[\mu_0 V_k(0, x_2+1) + 2\mu_1 V_k(1, x_2+1)]\\
+&\mu_0[\mu_0 V_k(0, x_2-1) + 2\mu_1 V_k(0, x_2)]\\
+&2\mu_1[\mu_0 V_k(0, x_2) + 2\mu_1 V_k(1, x_2)].
\end{aligned}$$

Again, in the each bracket—but the third—we can use the assumed property of V_k for $x_1, x_2 > 0$, and leave the third term untouched, which gives:

$$\begin{aligned}
=&\mu_0[\lambda_1 V_k(2, x_2-1) + \lambda_2 V_k(1, x_2) + \mu_0 V_k(0, x_2-1) + 2\mu_1 V_k(1, x_2-1)]\\
+&2\mu_1[\lambda_1 V_k(2, x_2) + \lambda_2 V_k(1, x_2+1) + \mu_0 V_k(0, x_2) + 2\mu_1 V_k(1, x_2)],
\end{aligned}$$

which by the assumed property of V_k for $x_1, x_2 > 0$ is finally equal to

$$\mu_0 V_{k+1}(1, x_2-1) + 2\mu_1 V_{k+1}(1, x_2).$$

The case $x_2 = 1, x_1 > 0$ is similar, and the case $x_1, x_2 = 1$ is included in the latter. $\square$

We deduce from the previous propositions that any policy that never uses both servers in parallel (when $x_1, x_2 > 0$), and serves only the non empty queue, when one of them is empty, is stochastically optimal.

Note that in the special case $2c_1 = c_0$, the inequality in Proposition 4 is in fact an equality, and the terms associated to the three actions are all equal. This implies that any policy—provided it serves only the non empty queue when one is empty—is optimal. Of course this is a rather theoretical situation, very unlikely in reality.

4.4 Stability conditions

When $2c_1 \leq c_0$, only one queue is served at a time, and the service rate is constant equal to μ_0, so in order to be stable, the total arrival rate must be smaller than μ_0, giving:

$$\lambda_1 + \lambda_2 < \mu_0,$$

as indicated in Figure 2.

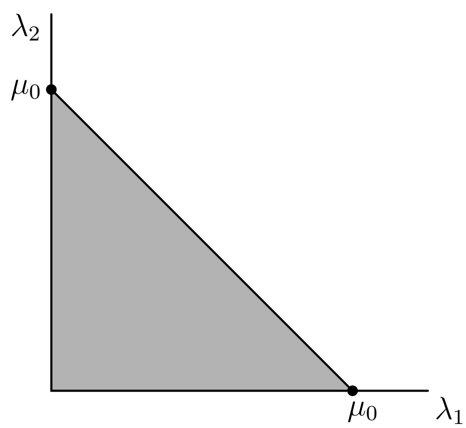

Figure 2: Stability region, $2c_1 \leq c_0$

When $2c_1 > c_0$, the model becomes a coupled processors model, and its stability conditions are as follows:

$$\min\left(\frac{\lambda_1}{\mu_1}, \frac{\lambda_2}{\mu_1}\right) < 1 \tag{4.2}$$

$$\text{if } \frac{\lambda_i}{\mu_1} < 1, \quad \lambda_j < \frac{\lambda_i}{\mu_1}\mu_1 + (1 - \frac{\lambda_i}{\mu_1})\mu_0, \quad j \neq i, \tag{4.3}$$

as proved in [11]. An intuitive interpretation of these conditions is indeed that if condition (4.2) is not met, both queues would grow unbounded when served at the same time according to the optimal policy. Condition (4.2) ensures that at least one queue i will be emptied regularly. Consequently, this queue i, which has a load $\frac{\lambda_i}{\mu_1}$ will be non empty for a fraction $\frac{\lambda_i}{\mu_1}$ of the time. When queue i is non empty, queue j gets a service rate of μ_1, and μ_0 otherwise. The average service rate for queue j is then $\frac{\lambda_i}{\mu_1}\mu_1 + (1 - \frac{\lambda_i}{\mu_1})\mu_0$, and condition (4.3) ensures its stability.

These conditions reduce to a piecewise linear stability region:

Suppose that $\lambda_1 < \mu_1$. Then the first condition is met, and the second implies

$$\lambda_2 < \frac{\lambda_1}{\mu_1}\mu_1 + (1 - \frac{\lambda_1}{\mu_1})\mu_0$$
$$\lambda_2 < \mu_0 - \lambda_1 \frac{\mu_0 - \mu_1}{\mu_1}.$$

If on the contrary, $\lambda_1 > \mu_1$, then the first condition implies $\lambda_2 < \mu_1$ and therefore the second condition gives

$$\lambda_1 < \frac{\lambda_2}{\mu_1}\mu_1 + (1 - \frac{\lambda_2}{\mu_1})\mu_0$$
$$\lambda_2 < \mu_1 \frac{\mu_0 - \lambda_1}{\mu_0 - \mu_1},$$

finally giving the stability region in Figure 3.

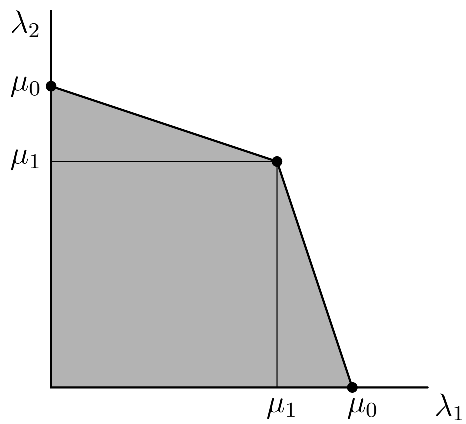

Figure 3: Stability region, $2c_1 > c_0$

4.5 Numerical considerations

Given the recursive definition of the V_k functions, it is possible to compute them numerically without approximation (other than the loss of precision due to the inherent use of floating point representation). It can be however numerically intensive for big values of k. Indeed, we see from the definition that computing $V_k(x_1, x_2)$ will require computing the values of $V_{k-1}(x_1 \pm 1, x_2 \pm 1)$ and $V_{k-1}(x_1, x_2)$, and so on down to $k = 0$ where the values are read directly as $\mathbf{1}_{(x_1+x_2>s)}$.

Even using memoization techniques, computing $V_k(x_1, x_2)$ requires computing first a number of values that grows as $O(k^3)$ (see Figure 4). The exact value is in fact $\frac{2k^3+k}{3}$, as proved in Appendix A.

Furthermore, to evaluate the performance of the policies, it is interesting to get the distribution of the total number of users in the system, for instance when starting from the empty state. Adding the index s to the definition of the V_k functions

$$V_{k,s}(x) = P[N_1(k) + N_2(k) > s | N(0) = x],$$

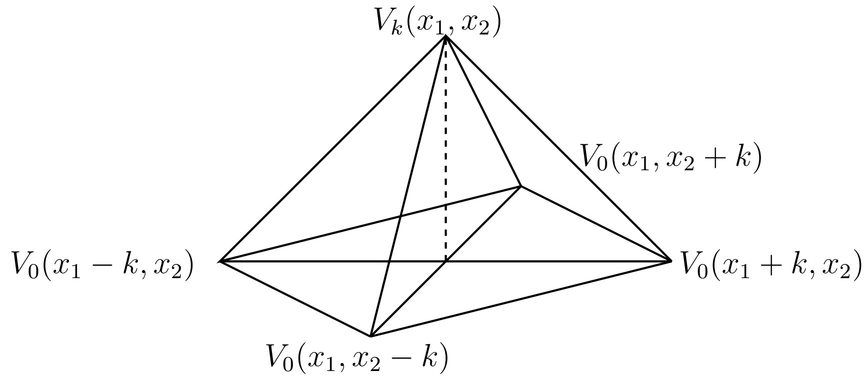

Figure 4: Pyramid of values required to compute $V_k(x_1, x_2)$

we can write

$$P[N_1(k) + N_2(k) = s | N(0) = (0, 0)] = V_{k,s-1}(0, 0) - V_{k,s}(0, 0)$$

and from this compute the distribution of N for any k. Note however that for each new s, the values must be recomputed over the entire state space, which is again quite computationally expensive.

If the system is stable, the steady-state distribution is obtained by making k tend to infinity. Then the dependency on the initial state (here taken to be $(0, 0)$) disappears. Again, it is numerically impossible to truly make k go to infinity, so some kind of compromise between convergence and computational limitations must be defined. We can for instance check the behavior of $P[N = 0]$. If the system is stable, it should converge towards a finite positive value. If instead it is unstable, it will converge towards 0.

Low interference, stable case We choose a set of parameters that place us in the case $2\mu_1 > \mu_0$, and that comply with the stability conditions:

$$\mu_0 = 1.2, \mu_1 = 0.9, \lambda_1 = 0.2, \lambda_2 = 0.3.$$

Figures 5 and 6 give the values of $V_{40,4}$ and of the distribution $P[N = x]$ for a high value of k, which ensured its convergence. As expected, $P[N = 0]$ is positive, and the distribution quickly decreases, since the chosen load is quite small.

For comparison purposes, Figure 7 shows the difference between $V_{40,4}$ and the same function but taken for a non-optimal policy (in this particular example, serving only one class at a time). We can see as expected that the difference is always negative. The plot has similar shapes for different values of s, illustrating that $V_k(x) = P[N_1(k) + N_2(k) > s | N(0) = x]$ is minimal over the set of policies for all s, and that the previously described policy is indeed stochastically optimal.

Low interference, unstable case We now keep the same system parameters, but choose a load that lies outside the stability region:

$$\lambda_1 = 0.8, \quad \lambda_2 = 1.$$

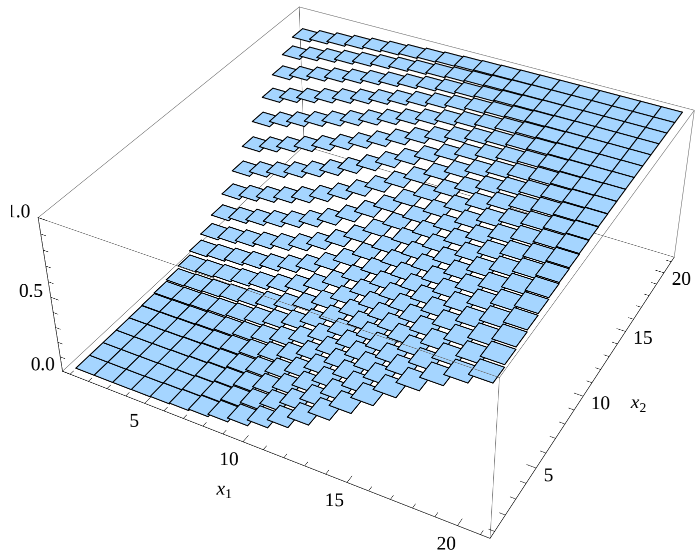

Figure 5: $V_{40,4}$, stable low interference case

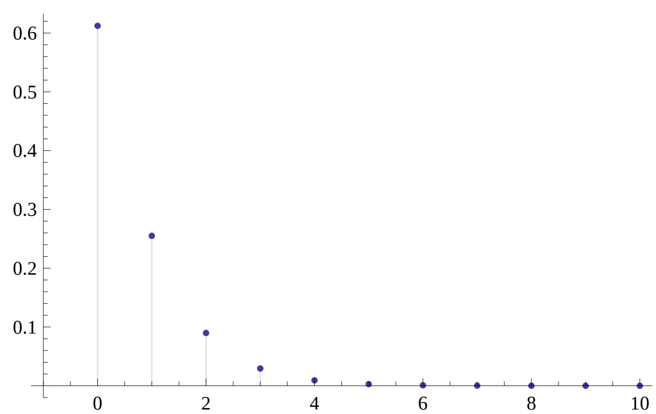

Figure 6: Probability distribution of N (taken after convergence, at $k \simeq 80$)

The distribution of N (shown in Figure 8) does not converge anymore: the steady state distribution does not exist, the system is unstable and N will grow unbounded when $k \to \infty$.

High interference, stable case Now considering $2\mu_1 \leq \mu_0$, we experiment with the following parameters:

$$\mu_0 = 1.2, \mu_1 = 0.4, \lambda_1 = 0.4, \lambda_2 = 0.2.$$

The V_k functions have roughly the same shape as before (see Figure 9), with the specificity that they are now symmetrical. In fact, this result is even stronger, and comes from Property 5:

$$V_k(x_1, x_2 - 1) = V_k(x_1 - 1, x_2)$$

implies that V_k is constant for $x_1 + x_2$ constant. In particular, $V_k(x_1, x_2) = V_k(x_2, x_1)$.

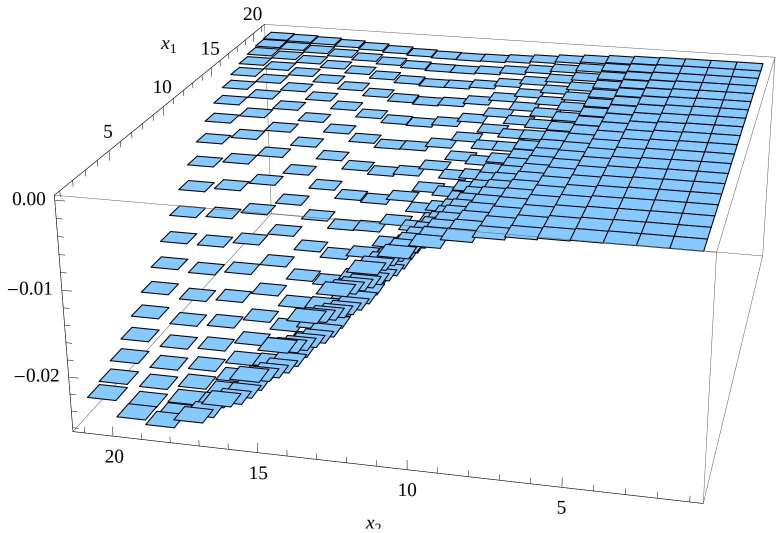

Figure 7: $V_{40,4} - V_{40,4}^{\text{non-opt}}$

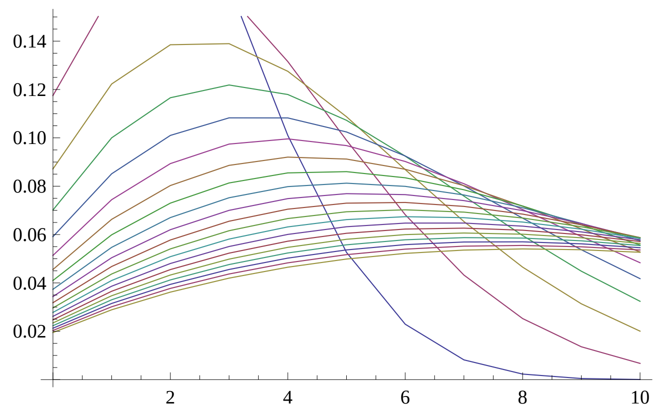

Figure 8: Probability distribution of N, for $k \in \{10, 20, \ldots, 200\}$

4.6 Direct Markov process solving

With the assumptions of Poisson arrivals and exponentially distributed service requirements, the system is in fact a 2-dimensional Markov process, and the steady state distribution can be found by solving its global balance equations.

We have seen that, for instance in low interference, the policy is to serve both queues when they are non empty, and to serve only the non empty queue when one of them is empty. This leads to a state transition diagram as in Figure 10. Of course, this diagram shows truncation of the state space, which is necessary to be able to do any numerical computation.

The next step is the construction of the associated state transition matrix. We number the states arbitrarily (here in lexicographic order) and build the matrix $\mathbf{Q}$ where q_{ij} is the transition rate from state i to state j and $q_{ii} = -\sum_{j \neq i} q_{ij}$. Since each state is neighbor to a maximum of only four other states, $\mathbf{Q}$ is a sparse matrix. For instance, truncation to $3^2 = 9$ states gives the following transition matrix (optimal

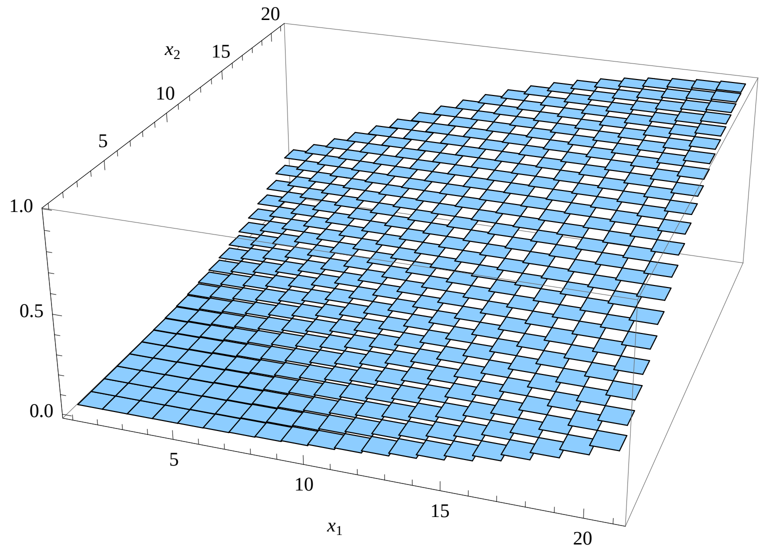

Figure 9: V_{80}^{4}, stable high interference case

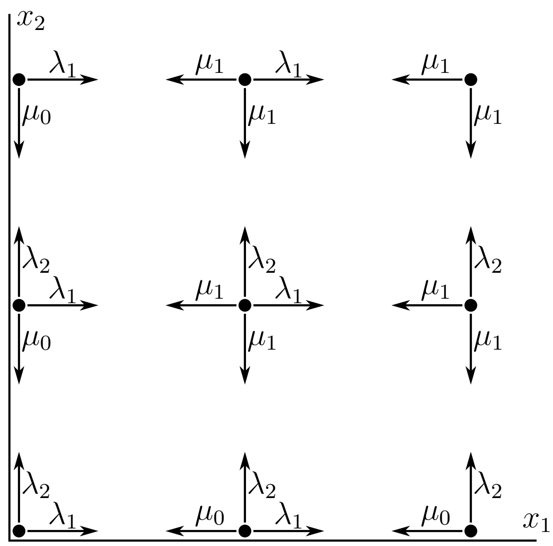

Figure 10: State transition diagram for the optimal policy in low interference case

policy in low interference case):

$$\begin{pmatrix}
-\lambda_1-\lambda_2 & \lambda_2 & 0 & \lambda_1 & 0 & 0 & 0 & 0 & 0 \\
\mu_0 & -\lambda_1-\lambda_2-\mu_0 & \lambda_2 & 0 & \lambda_1 & 0 & 0 & 0 & 0 \\
0 & \mu_0 & -\mu_0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\mu_0 & 0 & 0 & -\lambda_1-\lambda_2-\mu_0 & \lambda_2 & 0 & \lambda_1 & 0 & 0 \\
0 & \mu_1 & 0 & \mu_1 & -\lambda_1-\lambda_2-2\mu_1 & \lambda_2 & 0 & \lambda_1 & 0 \\
0 & 0 & \mu_1 & 0 & \mu_1 & -2\mu_1 & 0 & 0 & 0 \\
0 & 0 & 0 & \mu_0 & 0 & 0 & -\mu_0 & 0 & 0 \\
0 & 0 & 0 & 0 & \mu_1 & 0 & \mu_1 & -2\mu_1 & 0 \\
0 & 0 & 0 & 0 & 0 & \mu_1 & 0 & \mu_1 & -2\mu_1
\end{pmatrix}$$

As proved in Section 3.1, the steady state distribution $\boldsymbol{\pi}$ is found by solving

$$\boldsymbol{\pi}^T \mathbf{Q} = \mathbf{0},$$

effectively solving the global balance equations of the Markov process, and by normalizing $\boldsymbol{\pi}$ to comply with the normalizing condition $\sum_i \pi_i = 1$.

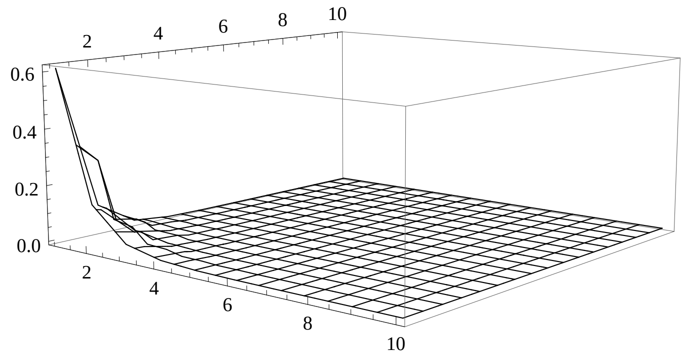

Figure 11: Steady state distribution for $\lambda_1 = 0.2$, $\lambda_2 = 0.3$ (stable case)

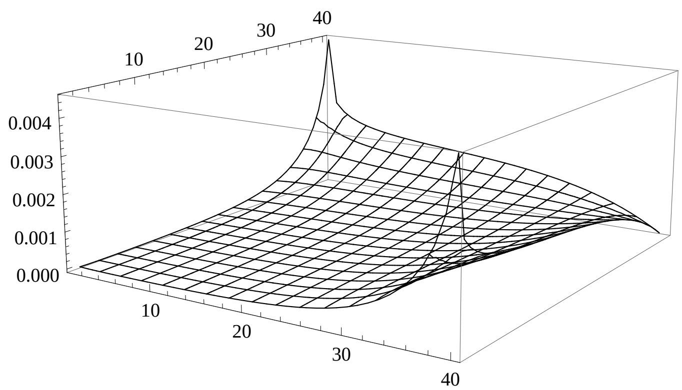

Figure 12: Steady state distribution for $\lambda_1 = 1.0$, $\lambda_2 = 1.0$ (unstable case)

The truncation of the state space has evidently an impact on the accuracy of the resulting distribution. In the following examples, we found that 40^2 states were enough to reach convergence in the stable cases, while still being small enough to be computed in a reasonable time.

Note however that a real-life system never has infinite waiting places. When one queue is full, the incoming user is dropped. If the system size is known, the truncation can be adjusted to it, and the numerical approximation will be actually closer to reality. Here we nevertheless choose a system big enough to appear as infinite in the stable cases.

Figures 11 and 12 show results for a stable and unstable case, with the same low interference system parameters as before: $\mu_0 = 1.2$ and $\mu_1 = 0.9$.

We can note that in the unstable case, the high probabilities accumulate towards the upper boundaries of the state space, implying a high average number of users, and a lot of refused connections.

In order to compare with the results of the previous method of calculation, we can derive from these 2-dimensional distributions the distribution of the total number of users in the system, by simply adding the probabilities on the $x_1 + x_2 = n$ diagonals. In this case we obtain the distribution represented in Figure 13, strictly equal to the one found earlier shown in Figure 6.

Finally, to illustrate the optimality of this policy $\tilde{\pi}$, we compute by the same

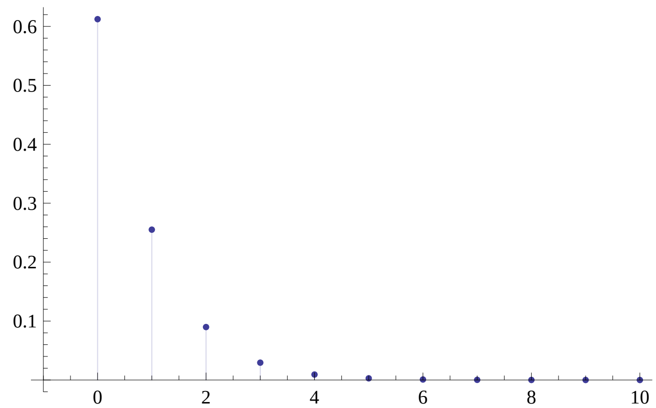

Figure 13: Steady state distribution of the total number of users for $\lambda_1 = 0.2$, $\lambda_2 = 0.3$ (stable case)

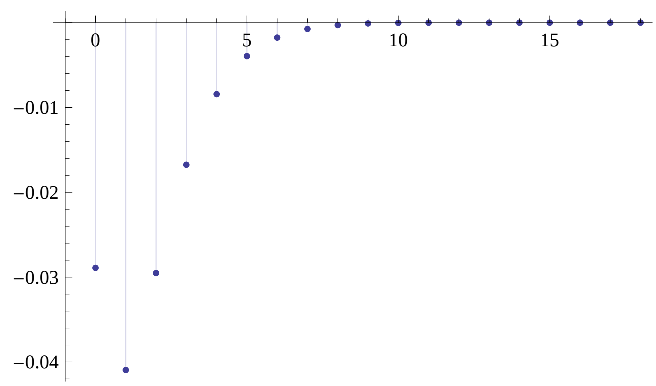

Figure 14: $P[N^{\pi} \leq x] - P[N^{\tilde{\pi}} \leq x]$

means the steady state distribution associated with another—non optimal—policy π. For instance we use a policy that always serves only one queue at a given time. From the resulting distribution we get the cumulative distribution function, that we compare to the one associated with the optimal policy (see Figure 14).

We note that $P[N^{\pi} \leq x] \leq P[N^{\tilde{\pi}} \leq x]$, which implies

$$P[N^{\tilde{\pi}} > x] \leq P[N^{\pi} > x], \quad \forall x \geq 0.$$

This was expected, since this is a direct consequence of definition (4.1) (stochastic optimality of the policy π).

5 Policy improvement

In this section we focus on obtaining an optimal policy by application of the Markov Decision Process theory. We first derive the optimal static policy, and then experiment with one step improvement.

5.1 Optimal static policy, low interference

We use the same model as in Section 4. In addition to the three main modes of serving $(0, c_0)$, $(c_0, 0)$ and (c_1, c_1), any convex combination of the three can be reached in a time-division fashion, with sufficient granularity to be considered continuous. By also considering turning off the two stations at the same time (giving $(0, 0)$ as the rate vector), we obtain the rate region in Figure 15. We also assume for the time being that $2c_1 > c_0$ (previously called "low interference" case).

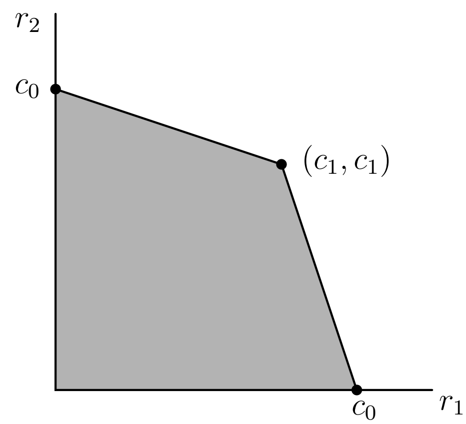

Figure 15: Achievable rate region

In a static policy (that is, where r_1 and r_2 do not vary over time), the system reduces to two independent M/M/1 queues, with parameters $(\lambda_1, r_1/E[X])$ and $(\lambda_2, r_2/E[X])$. Naturally, to optimize the performance we consider minimizing the average lengths of the queues. An M/M/1 queue with parameters (λ, μ) has an average number of users given by equation (3.3) (proved in Section 3.2):

$$E[N] = \frac{\lambda}{\mu - \lambda}.$$

Therefore for a rate vector of (r_1, r_2), the number of users in the two corresponding independent queues is simply their sum:

$$\bar{N} = \frac{\lambda_1}{r_1/E[X] - \lambda_1} + \frac{\lambda_2}{r_2/E[X] - \lambda_2}.$$

To clarify the notations, we will from now on use $\bar{X}$ as the average file size, giving the expression:

$$\bar{N} = \frac{\lambda_1 \bar{X}}{r_1 - \lambda_1 \bar{X}} + \frac{\lambda_2 \bar{X}}{r_2 - \lambda_2 \bar{X}}.$$

For any point (r_1, r_2) strictly within the rate region, increasing r_1 or r_2 will decrease $\bar{N}$. We can therefore deduce that the optimal point will be on the upper boundary of the rate region, and assume the following relation:

$$r_2 = \begin{cases} c_0 - \frac{c_0 - c_1}{c_1} r_1, & r_1 \leq c_1 \\ \frac{c_1}{c_0 - c_1}(c_0 - r_1), & r_1 \geq c_1 \end{cases}.$$

The rate vector (r_1, r_2) and consequently the value of $\bar{N}$ are now functions of the single variable r_1, which greatly simplifies the analysis.

Stability Since the system is composed of two independent M/M/1 queues, they should each be stable, which is ensured by $\lambda_i < r_i/\bar{X}$. We therefore have the following stability condition:

$$r_1 > \lambda_1 \bar{X} \text{ and } r_2 > \lambda_2 \bar{X}.$$

Interestingly, this results in the same stability region for λ_1 and λ_2 than previously derived for the stochastically optimal policy, where the queues were not independent but coupled (conditions (4.2) and (4.3), Figure 3). Fixing the r_i gives the stability region shown in Figure 16. On the other hand, fixing the arrival rates restricts r_1 further (Figure 17).

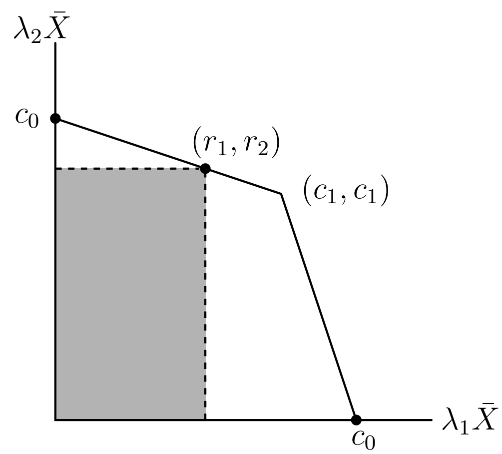

Figure 16: Stability region of (λ_1, λ_2), for (r_1, r_2) fixed

In order to get the optimal static policy, we need to find the minimum of $\bar{N}$ over the stable range for r_1. In the following paragraphs we divide the problem in several cases, based on the relative values of the λ_i. Furthermore, without loss of generality we assume that $\lambda_1 < \lambda_2$.

Case $\lambda_2 \geq \mu_1$ See Figure 18. In this situation, $r_1 \leq c_1$, which implies

$$r_2 = c_0 - \frac{c_0 - c_1}{c_1} r_1.$$

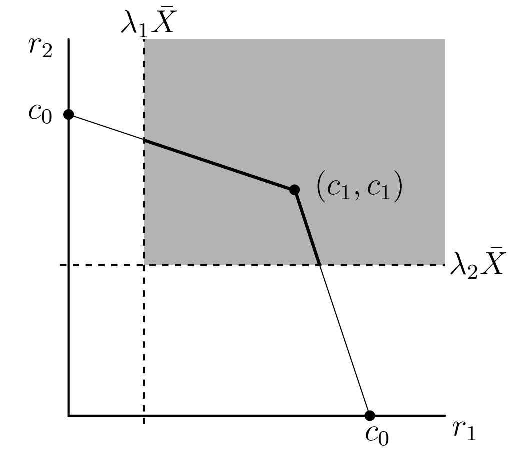

Figure 17: Stability region of (r_1, r_2), for (λ_1, λ_2) fixed

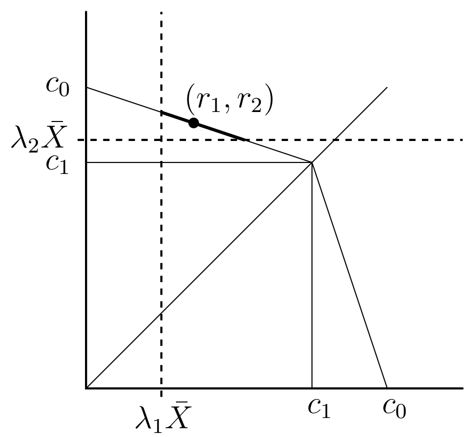

Figure 18: Stable range for (r_1, r_2) in case $\lambda_2 \geq \mu_1$

Given the definition of $\bar{N}$, the latter will have vertical asymptotes at both end of the stability range, when

$$r_1 \to \lambda_1\bar{X} \text{ and } r_2 \to \lambda_2\bar{X} \iff r_1 \to r_2^{-1}(\lambda_2\bar{X}) \iff r_1 \to (c_0 - \lambda_2\bar{X})\frac{c_1}{c_0 - c_1}.$$

Since in the stability range, $\bar{N}$ is otherwise positive and continuous, we can expect a plot similar to Figure 19 in which the minimum can be found by setting the derivative to zero.

$$\begin{aligned}\frac{\partial \bar{N}}{\partial r_1} =& \frac{-\lambda_1\bar{X}}{(r_1 - \lambda_1\bar{X})^2} + \frac{-\lambda_2\bar{X}}{(r_2 - \lambda_2\bar{X})^2}\frac{\partial r_2}{\partial r_1} = 0 \\ & \frac{\lambda_1\bar{X}}{(r_1 - \lambda_1\bar{X})^2} = \frac{\lambda_2\bar{X}}{(r_2 - \lambda_2\bar{X})^2}\frac{c_0 - c_1}{c_1}.\end{aligned}$$

Since we know that $r_1 > \lambda_1\bar{X}$ and $r_2 > \lambda_2\bar{X}$, we can directly simplify the second degree equation by taking the square root of each side, resulting in a single solution

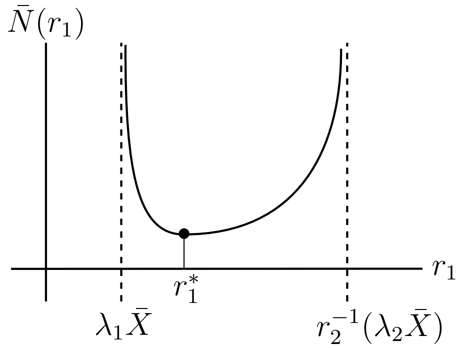

Figure 19: $\bar{N}$ in case $\lambda_2 \geq \mu_1$

(meaning that the second one is outside the stability range):

$$\begin{aligned}
\frac{\sqrt{\lambda_1\bar{X}}}{r_1-\lambda_1\bar{X}} &= \frac{\sqrt{\lambda_2\bar{X}}}{r_2-\lambda_2\bar{X}}\sqrt{\frac{c_0-c_1}{c_1}} \\
(r_2-\lambda_2\bar{X})\sqrt{\lambda_1\bar{X}} &= (r_1-\lambda_1\bar{X})\sqrt{\lambda_2\bar{X}\frac{c_0-c_1}{c_1}} \\
(c_0-\frac{c_0-c_1}{c_1}r_1-\lambda_2\bar{X})\sqrt{\lambda_1\bar{X}} &= (r_1-\lambda_1\bar{X})\sqrt{\lambda_2\bar{X}\frac{c_0-c_1}{c_1}} \\
(c_0-\lambda_2\bar{X})\sqrt{\lambda_1\bar{X}}+\lambda_1\bar{X}\sqrt{\lambda_2\bar{X}\frac{c_0-c_1}{c_1}} &= r_1\left(\sqrt{\lambda_2\bar{X}\frac{c_0-c_1}{c_1}}+\frac{c_0-c_1}{c_1}\sqrt{\lambda_1\bar{X}}\right).
\end{aligned}$$

Finally, the unique solution for $\frac{\partial\bar{N}}{\partial r_1}=0$ within the stability range is given by:

$$r_1^* = \frac{(c_0-\lambda_2\bar{X})\sqrt{\lambda_1\bar{X}}+\lambda_1\bar{X}\sqrt{\lambda_2\bar{X}\frac{c_0-c_1}{c_1}}}{\sqrt{\lambda_2\bar{X}\frac{c_0-c_1}{c_1}}+\frac{c_0-c_1}{c_1}\sqrt{\lambda_1\bar{X}}}. \tag{5.1}$$

Note that by rewriting

$$\begin{aligned}
r_1^* &= \frac{(c_0-\lambda_2\bar{X})\sqrt{\lambda_1\bar{X}}+\lambda_1\bar{X}\sqrt{\lambda_2\bar{X}\frac{c_0-c_1}{c_1}}}{\sqrt{\lambda_2\bar{X}\frac{c_0-c_1}{c_1}}+\frac{c_0-c_1}{c_1}\sqrt{\lambda_1\bar{X}}}\cdot\frac{\frac{c_1}{c_0-c_1}}{\frac{c_1}{c_0-c_1}} \\
r_1^* &= \frac{(c_0-\lambda_2\bar{X})\frac{c_1}{c_0-c_1}\sqrt{\lambda_1\bar{X}}+\lambda_1\bar{X}\sqrt{\lambda_2\bar{X}\frac{c_1}{c_0-c_1}}}{\sqrt{\lambda_1\bar{X}}+\sqrt{\lambda_2\bar{X}\frac{c_1}{c_0-c_1}}},
\end{aligned}$$

we make appear the fact that r_1^* can be expressed as the average of the stability limits weighted by the coefficients $\sqrt{\lambda_1\bar{X}}$ and $\sqrt{\lambda_2\bar{X}\frac{c_1}{c_0-c_1}}$.

To conclude, in the case $\lambda_2 \geq \mu_1$, there is a unique optimal static policy, which consists in serving the queues with the fixed rates r_1^* and $r_2(r_1^*)$.

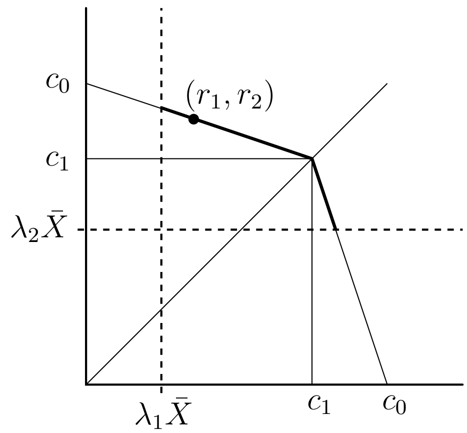

Figure 20: Stable range for (r_1, r_2) in case $\lambda_2 < \mu_1$

Case $\lambda_2 < \mu_1$ See Figure 20.

Numerical experiments—backed up by the following analysis—show that two situations can occur. Because $\lambda_2 < \mu_1$, the upper bound of the stability region $r_2^{-1}(\lambda_2\bar{X})$ is now greater than c_1. Given the piecewise definition of r_2 with a cut on c_1, $\bar{N}$ is now also a piecewise function over the stability region. As we will show, it can take two different shapes (Figure 21).

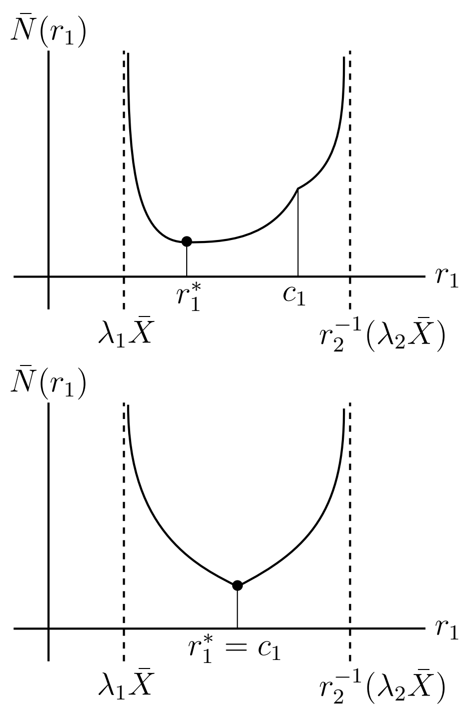

Figure 21: The two possibilities for $\bar{N}$ in case $\lambda_2 < \mu_1$

First we need to prove that $\bar{N}$ is strictly increasing on the "right" segment ($r_1 \geq c_1$). Then we will find the conditions that separate the two possible cases on the "left" segment ($r_1 \leq c_1$), as well as the resulting minima for $\bar{N}$.

Case $\lambda_2 < \mu_1$, right segment In this situation, $c_1 \le r_1 < r_2^{-1}(\lambda_2 \bar{X})$, which implies

$$r_2 = \frac{c_1}{c_0 - c_1}(c_0 - r_1).$$

We consider the derivative of $\bar{N}$:

$$\begin{aligned}\frac{\partial \bar{N}}{\partial r_1} &= \frac{-\lambda_1 \bar{X}}{(r_1 - \lambda_1 \bar{X})^2} + \frac{-\lambda_2 \bar{X}}{(r_2 - \lambda_2 \bar{X})^2}\frac{\partial r_2}{\partial r_1} \\ &= \frac{\lambda_2 \bar{X}}{(r_2 - \lambda_2 \bar{X})^2}\frac{c_1}{c_0 - c_1} - \frac{\lambda_1 \bar{X}}{(r_1 - \lambda_1 \bar{X})^2},\end{aligned}$$

and we therefore have:

$$\begin{aligned}\frac{\partial \bar{N}}{\partial r_1} > 0 &\iff \frac{\lambda_2 \bar{X}}{\lambda_1 \bar{X}}\frac{(r_1 - \lambda_1 \bar{X})^2}{(r_2 - \lambda_2 \bar{X})^2}\frac{c_1}{c_0 - c_1} > 1 \\ &\Longleftarrow \frac{\lambda_2 \bar{X}}{\lambda_1 \bar{X}} > 1 \text{ and } \frac{(r_1 - \lambda_1 \bar{X})^2}{(r_2 - \lambda_2 \bar{X})^2} > 1 \text{ and } \frac{c_1}{c_0 - c_1} > 1.\end{aligned}$$

The first inequality is verified directly, by the assumption $\lambda_1 < \lambda_2$.

Since $r_1 > \lambda_1 \bar{X}$ and $r_2 > \lambda_2 \bar{X}$, the second inequality reduces to

$$\begin{aligned}&\frac{r_1 - \lambda_1 \bar{X}}{r_2 - \lambda_2 \bar{X}} > 1 \\ &r_1 - \lambda_1 \bar{X} > r_2 - \lambda_2 \bar{X} \\ &(r_1 - r_2) + (\lambda_2 - \lambda_1)\bar{X} > 0\end{aligned}$$

which is verified, since $\lambda_1 < \lambda_2$ and $r_1 \ge c_1 \ge r_2$.

Finally, $2c_1 > c_0$ implies $c_1 > c_0 - c_1$ and as expected,

$$\frac{c_1}{c_0 - c_1} > 1.$$

We have proved that $\bar{N}$ is strictly increasing on $[c_1, r_2^{-1}(\lambda_2 \bar{X})[$, which implies that its minimum on this segment is reached at $r_1 = c_1$.

Case $\lambda_2 < \mu_1$, left segment We now have $\lambda_1 \bar{X} < r_1 \le c_1$. Consider the following: prolong the "left" piece of $\bar{N}$ into $\bar{N}_{left}$ by extending its definition from $]\lambda_1 \bar{X}, c_1]$ to the r_1 greater than c_1 as well.

In other words, let

$$r_{2,left} = c_0 - \frac{c_0 - c_1}{c_1} r_1, \quad r_1 \in [0, c_0]$$

and

$$\bar{N}_{left} = \frac{\lambda_1 \bar{X}}{r_1 - \lambda_1 \bar{X}} + \frac{\lambda_2 \bar{X}}{r_{2,left} - \lambda_2 \bar{X}}.$$

Then as before (case $\lambda_2 \geq \mu_1$, page 31) it reaches its minimum at r_1^*, previously defined in equation (5.1):

$$r_1^* = \frac{(c_0 - \lambda_2\bar{X})\sqrt{\lambda_1\bar{X}} + \lambda_1\bar{X}\sqrt{\lambda_2\bar{X}\frac{c_0-c_1}{c_1}}}{\sqrt{\lambda_2\bar{X}\frac{c_0-c_1}{c_1}} + \frac{c_0-c_1}{c_1}\sqrt{\lambda_1\bar{X}}}.$$

The relevant question is now to find out whether $r_1^* < c_1$ or $r_1^* \geq c_1$. Indeed, in this first case, $\bar{N}(r_1^*) < \bar{N}(c_1)$ and therefore $\bar{N}$ reaches its minimum in r_1^*. In the second case, $\bar{N}$ is minimal at c_1 (see Figure 22).

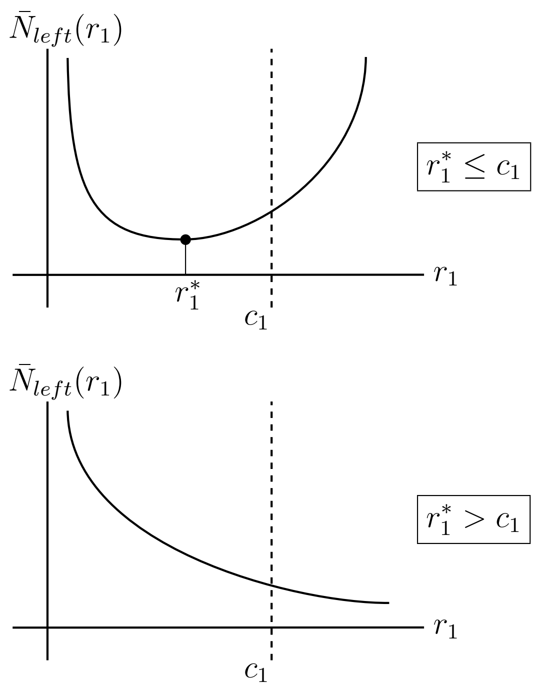

Figure 22: The two possibilities for $\bar{N}_{left}$ in case $\lambda_2 < \mu_1$

Comparing directly r_1^* to c_1 is difficult and leads to an expression that does not give any insight on the structure of the problem. Instead, we will note that since $\bar{N}_{left}$ is continuous, has only one minimum on the considered interval, and goes to infinity at $\lambda_1\bar{X}$ and $r_{2,left}^{-1}(\lambda_2\bar{X})$, it is necessarily decreasing before r_1^* and increasing after. Therefore,

$$r_1^* < c_1 \iff \left.\frac{\partial \bar{N}_{left}}{\partial r_1}\right|_{r_1=c_1} > 0.$$

$$\begin{aligned}\left.\frac{\partial \bar{N}_{left}}{\partial r_1}\right|_{r_1=c_1} &= \frac{\lambda_2\bar{X}}{(r_2(c_1)-\lambda_2\bar{X})^2}\frac{c_0-c_1}{c_1} - \frac{\lambda_1\bar{X}}{(c_1-\lambda_1\bar{X})^2} \\ &= \frac{\lambda_2\bar{X}}{(c_1-\lambda_2\bar{X})^2}\frac{c_0-c_1}{c_1} - \frac{\lambda_1\bar{X}}{(c_1-\lambda_1\bar{X})^2},\end{aligned}$$

and finally

$$r_1^* < c_1 \iff \lambda_2\bar{X}(c_1 - \lambda_1\bar{X})^2\frac{c_0 - c_1}{c_1} > \lambda_1\bar{X}(c_1 - \lambda_2\bar{X})^2$$
$$\iff \lambda_2(\mu_1 - \lambda_1)^2\frac{\mu_0 - \mu_1}{\mu_1} > \lambda_1(\mu_1 - \lambda_2)^2. \tag{5.2}$$

Although the boundary in the form $\lambda_2(\lambda_1)$ exists, and is given by solving the quadratic equation

$$\lambda_2(\mu_1 - \lambda_1)^2\frac{\mu_0 - \mu_1}{\mu_1} = \lambda_1(\mu_1 - \lambda_2)^2,$$

it has little interest. We can however have an overview of its properties by noticing that

- $\lambda_2(0) = 0$
- $\lambda_2(\mu_1) = \mu_1$
- $\lambda_2(\lambda_1)$ is strictly increasing on the interval $[0, \mu_1]$.

Proof. The first two statements are proved by simple value substitution:

$$\lambda_2\mu_1(\mu_0 - \mu_1) = 0 \implies \lambda_2 = 0,$$
$$0 = \mu_1(\mu_1 - \lambda_2)^2 \implies \mu_1 - \lambda_2 = 0 \implies \lambda_2 = \mu_1.$$

The third by differentiating both sides of the equation with respect to λ_1:

$$\frac{\mu_0 - \mu_1}{\mu_1}\frac{\partial}{\partial\lambda_1}\left\{\lambda_2(\mu_1 - \lambda_1)^2\right\} = \frac{\partial}{\partial\lambda_1}\left\{\lambda_1(\mu_1 - \lambda_2)^2\right\}$$
$$\frac{\mu_0 - \mu_1}{\mu_1}\left(\frac{\partial\lambda_2}{\partial\lambda_1}(\mu_1 - \lambda_1)^2 - 2\lambda_2(\mu_1 - \lambda_1)\right) = (\mu_1 - \lambda_2)^2 - 2\lambda_1(\mu_1 - \lambda_2)\frac{\partial\lambda_2}{\partial\lambda_1}$$
$$\frac{\partial\lambda_2}{\partial\lambda_1}\cdot\left(\frac{\mu_0 - \mu_1}{\mu_1}(\mu_1 - \lambda_1)^2 + 2\lambda_1(\mu_1 - \lambda_2)\right) = (\mu_1 - \lambda_2)^2 + 2\lambda_2(\mu_1 - \lambda_1)\frac{\mu_0 - \mu_1}{\mu_1},$$

which gives

$$\frac{\partial\lambda_2}{\partial\lambda_1} = \frac{(\mu_1 - \lambda_2)^2 + 2\lambda_2(\mu_1 - \lambda_1)\frac{\mu_0 - \mu_1}{\mu_1}}{\frac{\mu_0 - \mu_1}{\mu_1}(\mu_1 - \lambda_1)^2 + 2\lambda_1(\mu_1 - \lambda_2)},$$

which is positive, since $0 < \lambda_1 < \lambda_2 < \mu_1 < \mu_0$. □

The closed region between this line and the $\lambda_2 = \lambda_1$ line then corresponds to the values of λ_i for which $r_1^* \geq c_1$, and therefore the minimum of $\bar{N}$ is reached for c_1.

Conclusion on the optimal static policy, low interference In light of the analysis of the three previous cases, and by taking in account the symmetry of the λ_i (we assumed $\lambda_1 < \lambda_2$, but for the opposite case a simple renaming of the variables brings them back in the desired order), the solution of the problem falls down into two cases, as illustrated in Figure 23:

- λ_1 and λ_2 are "close enough", as defined by inequation (5.2). The optimal static policy is to serve both queues with rate c_1 at all times.
- λ_1 and λ_2 are not close enough. The optimal static policy is to serve the queues with rates r_1^*—defined in equation (5.1)—and $r_2(r_1^*)$ at all times.

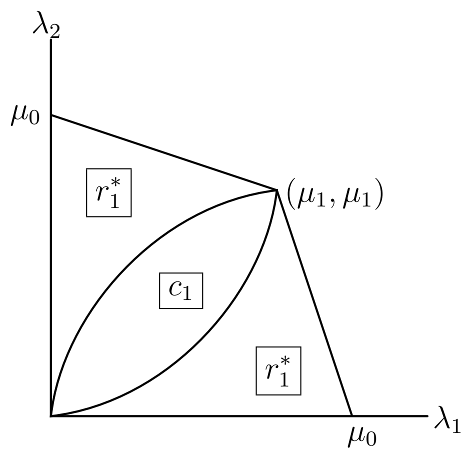

Figure 23: Regions for the optimal value of r_1, depending on (λ_1, λ_2)

5.2 Optimal static policy, high interference

We now consider the so called "high interference" case, that is when $2c_1 \leq c_0$. The achievable rate region is again the convex hull of the points $(0, c_0)$, $(c_0, 0)$, (c_1, c_1) and $(0, 0)$, as shown in Figure 24.

Since (c_1, c_1) is inside the rate region, we can already infer that using this mode will not be of any interest, and that the optimal policy will make use only of the single-queue modes. Formally, we derive as before the average number of users in the system as the sum of the lengths of two independent M/M/1 queues:

$$\bar{N} = \frac{\lambda_1 \bar{X}}{r_1 - \lambda_1 \bar{X}} + \frac{\lambda_2 \bar{X}}{r_2 - \lambda_2 \bar{X}}.$$

Since $\bar{N}$ is strictly decreasing in r_1 and r_2, we know its minimum will be on the upper boundary of the rate region, and we therefore assume:

$$r_2 = c_0 - r_1.$$

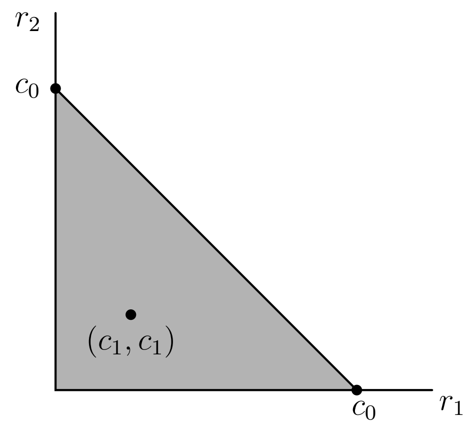

Figure 24: Achievable rate region, high interference

Stability As before, the stability of the system is ensured by the stability of the two independent queues:

$$r_1 > \lambda_1 \bar{X} \text{ and } r_2 > \lambda_2 \bar{X},$$

as illustrated in Figure 25. Stability is achieved when $r_1 \in]\lambda_1 \bar{X}, r_2^{-1}(\lambda_2 \bar{X})[$, that is for

$$\lambda_1 \bar{X} < r_1 < c_0 - \lambda_2 \bar{X}.$$

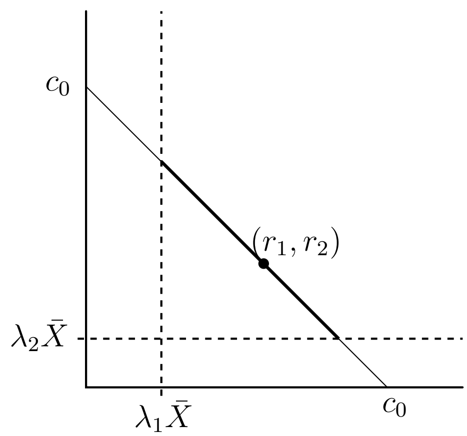

Figure 25: Stable range for (r_1, r_2)

Minimum We solve for the minimum of $\bar{N}$ as before:

$$\frac{\partial \bar{N}}{\partial r_1} = 0$$

$$\frac{-\lambda_1 \bar{X}}{(r_1 - \lambda_1 \bar{X})^2} + \frac{-\lambda_2 \bar{X}}{(c_0 - r_1 - \lambda_2 \bar{X})^2}(-1) = 0$$

$$\frac{\sqrt{\lambda_1 \bar{X}}}{r_1 - \lambda_1 \bar{X}} = \frac{\sqrt{\lambda_2 \bar{X}}}{c_0 - r_1 - \lambda_2 \bar{X}}$$

$$(c_0 - r_1 - \lambda_2 \bar{X})\sqrt{\lambda_1 \bar{X}} = (r_1 - \lambda_1 \bar{X})\sqrt{\lambda_2 \bar{X}}.$$

Finally,

$$r_1^* = \frac{(c_0 - \lambda_2 \bar{X})\sqrt{\lambda_1 \bar{X}} + \lambda_1 \bar{X}\sqrt{\lambda_2 \bar{X}}}{\sqrt{\lambda_1 \bar{X}} + \sqrt{\lambda_2 \bar{X}}}.$$

Note that:

- the result is independent of c_1. As long as c_1 is smaller than $c_0/2$, its actual value does not matter;
- again, r_1^* is in fact the average of the stability limits, weighted in this case with the coefficients $\sqrt{\lambda_i \bar{X}}$;
- since

  $$r_2^* = c_0 - r_1^* = \frac{(c_0 - \lambda_1 \bar{X})\sqrt{\lambda_2 \bar{X}} + \lambda_2 \bar{X}\sqrt{\lambda_1 \bar{X}}}{\sqrt{\lambda_1 \bar{X}} + \sqrt{\lambda_2 \bar{X}}},$$

  we can see that the results are symmetrical with respect to the λ_i: swapping λ_1 and λ_2 swaps r_1^* and r_2^*, which was to be expected, given the symmetry of the problem.

In fact the relation with the low interference case is even stronger. It is simply its limit as $c_1 \to c_0/2$. Indeed, as soon as c_1 becomes smaller than $c_0/2$, the (c_1, c_1) mode is never used anymore, and the problem stays equivalent to $c_1 = c_0/2$.

5.3 Policy improvement iteration

Starting from the optimal static policy, we can now apply policy improvement, as described in Section 3.2.

Initial policy We first need to find out the relative state values for the static policy. As stated previously, for the optimal static policy, the system is in fact two independent M/M/1 queues, with parameters $(\lambda_1, r_1^*/E[X])$ and $(\lambda_2, r_2(r_1^*)/E[X])$. We can therefore use directly the sum of relative state values for the two M/M/1 queues, derived in equation (3.2). With $\mu_{1s} = r_1^*/E[X]$ and $\mu_{2s} = r_2(r_1^*)/E[X]$:

$$v_s(n_1, n_2) = v_{M/M/1-\lambda_1,\mu_{1s}}(n_1) + v_{M/M/1-\lambda_2,\mu_{2s}}(n_2)$$
$$v_s(n_1, n_2) = \frac{1}{2}\frac{n_1(n_1+1)}{\mu_{1s} - \lambda_1} + \frac{1}{2}\frac{n_2(n_2+1)}{\mu_{2s} - \lambda_2}.$$

In order to alleviate further expressions, let us notice already that

$$v_s(n_1+1, n_2) - v_s(n_1, n_2) = \frac{n_1+1}{\mu_{1s} - \lambda_1}, \quad \forall n_1 \in \mathbb{N}$$

and

$$v_s(n_1, n_2+1) - v_s(n_1, n_2) = \frac{n_2+1}{\mu_{2s} - \lambda_2}, \quad \forall n_2 \in \mathbb{N}.$$

We can also derive immediately the cost rate

$$r_s(n_1, n_2) = n_1 + n_2$$

and the average cost rate

$$r_s = \frac{\lambda_1}{\mu_{1s} - \lambda_1} + \frac{\lambda_2}{\mu_{2s} - \lambda_2},$$

which are, again, simply the sums of the associated values in two independent M/M/1 queues.

First iteration To improve the policy, we now need to find for each state (n_1, n_2) the action a that will minimize the following expression, as stated in equation (3.1):

$$C_a(n_1, n_2) = r_s(n_1, n_2) - r_s + \sum_{(u,v)} q_{(n_1,n_2)\to(u,v)}(a) v_s(u, v).$$

Note that in this particular case, the cost rates are independent of the action. Only the transition rates vary. Also, the available actions are the same for each state, and correspond to the three service modes of the system: $(0, \mu_0)$, $(\mu_0, 0)$ and (μ_1, μ_1), denoted respectively "01", "10" and "11".

Furthermore, we take advantage of the definition of $q_{(n_1,n_2)\to(n_1,n_2)}$ to rewrite:

$$C_a(n_1, n_2) = r_s(n_1, n_2) - r_s + \sum_{(u,v)\neq(n_1,n_2)} q_{(n_1,n_2)\to(u,v)}(a) \left[v_s(u, v) - v_s(n_1, n_2)\right],$$

and we can finally write down the expressions for the three possible actions:

$$\begin{aligned} C_{10}(n_1, n_2) = r_s(n_1, n_2) - r_s &+ \lambda_1[v_s(n_1 + 1, n_2) - v_s(n_1, n_2)] \\ &+ \lambda_2[v_s(n_1, n_2 + 1) - v_s(n_1, n_2)] \\ &+ \mu_0[v_s((n_1 - 1)^+, n_2) - v_s(n_1, n_2)], \end{aligned}$$

$$\begin{aligned} C_{01}(n_1, n_2) = r_s(n_1, n_2) - r_s &+ \lambda_1[v_s(n_1 + 1, n_2) - v_s(n_1, n_2)] \\ &+ \lambda_2[v_s(n_1, n_2 + 1) - v_s(n_1, n_2)] \\ &+ \mu_0[v_s(n_1, (n_2 - 1)^+) - v_s(n_1, n_2)], \end{aligned}$$

$$\begin{aligned} C_{11}(n_1, n_2) = r_s(n_1, n_2) - r_s &+ \lambda_1[v_s(n_1 + 1, n_2) - v_s(n_1, n_2)] \\ &+ \lambda_2[v_s(n_1, n_2 + 1) - v_s(n_1, n_2)] \\ &+ \mu_1[v_s((n_1 - 1)^+, n_2) - v_s(n_1, n_2)] \\ &+ \mu_1[v_s(n_1, (n_2 - 1)^+) - v_s(n_1, n_2)]. \end{aligned}$$

We can already notice that the first four terms of each expression are independent of the chosen action. Let us regroup them under the constant K and expand the remaining terms:

$$
\begin{aligned}
C_{10}(n_1, n_2) &= K - \mu_0 \frac{n_1}{\mu_{1s} - \lambda_1}, \\
C_{01}(n_1, n_2) &= K - \mu_0 \frac{n_2}{\mu_{2s} - \lambda_2}, \\
C_{11}(n_1, n_2) &= K - \mu_1 \frac{n_1}{\mu_{1s} - \lambda_1} - \mu_1 \frac{n_2}{\mu_{2s} - \lambda_2}.
\end{aligned}
$$

Note that the special cases $n_1 = 0$ or $n_2 = 0$ are correctly included in the same expressions. We can also discard directly the case $(n_1, n_2) = (0, 0)$ which makes all three values equal: when both queues are empty, the service rates are irrelevant.

Comparing the three cases is now straightforward, and leads to the following equivalences:

$$
\begin{aligned}
C_{10}(n_1, n_2) > C_{01}(n_1, n_2) &\iff n_2 > n_1 \cdot \frac{\mu_{2s} - \lambda_2}{\mu_{1s} - \lambda_1}, \\
C_{11}(n_1, n_2) < C_{10}(n_1, n_2) &\iff n_2 > n_1 \cdot \frac{\mu_{2s} - \lambda_2}{\mu_{1s} - \lambda_1} \frac{\mu_0 - \mu_1}{\mu_1}, \\
C_{11}(n_1, n_2) < C_{10}(n_1, n_2) &\iff n_2 < n_1 \cdot \frac{\mu_{2s} - \lambda_2}{\mu_{1s} - \lambda_1} \frac{\mu_1}{\mu_0 - \mu_1},
\end{aligned}
$$

or with shortcut notations $R = \frac{\mu_{2s} - \lambda_2}{\mu_{1s} - \lambda_1}$, $S = \frac{\mu_0 - \mu_1}{\mu_1}$:

$$
\begin{aligned}
C_{10} > C_{01} &\iff n_2 > n_1 \cdot R, \\
C_{10} > C_{11} &\iff n_2 > n_1 \cdot RS, \\
C_{11} > C_{10} &\iff n_2 > n_1 \cdot R \frac{1}{S}.
\end{aligned}
$$

As before, the final result depends on the relative values of μ_1 and μ_0, the "high" and "low" interference cases.

High interference $2\mu_1 \leq \mu_0$

$$
\begin{aligned}
\mu_1 &\leq \mu_0 - \mu_1 \\
\frac{\mu_0 - \mu_1}{\mu_1} &\geq 1 \\
\implies S \geq 1, &\quad \frac{1}{S} \leq 1.
\end{aligned}
$$

We therefore have the order $\frac{1}{S} R \leq R \leq SR$, and two possible cases:

- $n_2 \leq n_1 R$. Consequently, $n_2 \leq n_1 SR$, which gives $C_{10} \leq C_{01}$ & $C_{10} \leq C_{11}$. The minimizing action is "10".
- $n_2 \geq n_1 R$. We have $n_2 \geq n_1 \frac{1}{S} R$, which gives $C_{10} \geq C_{01}$ & $C_{01} \leq C_{11}$. The minimizing action is "01".

The new policy is described by a linear switching curve of equation $n_2 = n_1 R$, as represented in Figure 26.

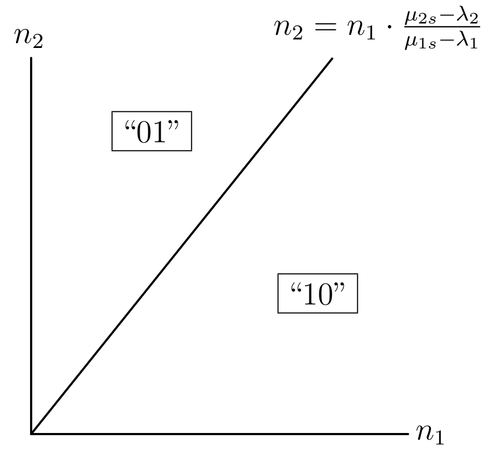

Figure 26: New policy, high interference

Low interference $2\mu_1 > \mu_0$

$$\mu_1 > \mu_0 - \mu_1$$
$$\frac{\mu_0 - \mu_1}{\mu_1} < 1$$
$$\Longrightarrow S < 1, \quad \frac{1}{S} > 1,$$

We therefore have the order $SR < R < \frac{1}{S}R$, and three possible cases:

- $n_2 < n_1 SR$. Consequently, $n_2 < n_1 R$, which gives $C_{10} < C_{11}$ & $C_{10} < C_{01}$. The minimizing action is "10".
- $n_1 SR < n_2 < n_1 \frac{1}{S} R$. Then $C_{10} > C_{11}$ & $C_{01} > C_{11}$. The minimizing action is "11".
- $n_2 > n_1 \frac{1}{S} R$. Then $n_2 > n_1 R$ and $C_{01} < C_{11}$ & $C_{01} < C_{10}$. The minimizing action is "01".

The new policy is described by two linear switching curves of equations $n_2 = n_1 SR$ and $n_2 = n_1 \frac{1}{S} R$, as represented in Figure 27.

Remarks on the first iteration

Optimality In the high interference case, the improved policy never makes use of the "11" mode, and is therefore stochastically optimal, according to Section 4.3. For the low interference case, however, one iteration is not enough to reach the stochastically optimal policy.

Initial policy Here we chose to take the optimal static policy as a starting point. However, it is in general not true that this will lead to the optimal first iterated policy. It is possible that non optimal starting points give better results. However, given the complexity of the expressions, it is unlikely to find explicitly an

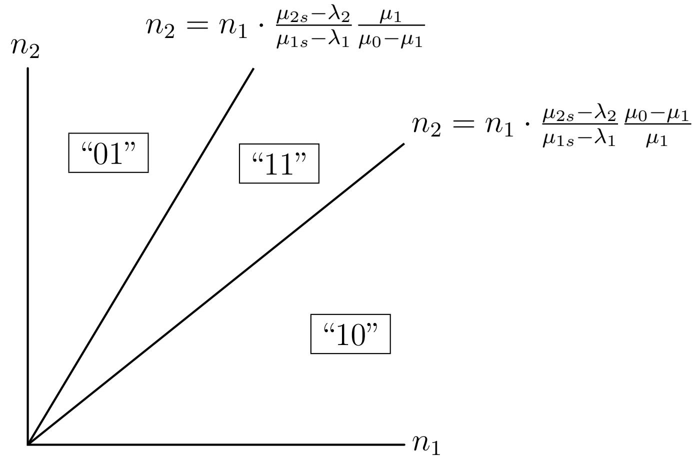

Figure 27: New policy, low interference

optimal starting point. Furthermore, the theory of Markov decision processes states that regardless of the starting policy, the average-optimal policy will be found at convergence. A "good" starting policy may only accelerate this convergence.

Improvement To quantify the actual improvement brought by the iteration, one would compute from the new policy (i.e. transition rates) the relative state values and average cost rate, using the Howard equation. However, now that the policy is described by switching curves, it is not static anymore, and the Howard equation takes a different form depending on high/low interference and the region in which the state is. This plus the fact that the state space is infinite makes it very unlikely to find a symbolic expression.

Following steps Again, given the description of the first iterated policy, it is difficult to consider applying one more improvement step symbolically, let alone applying it indefinitely until convergence. For given parameters, it is however almost trivial to do numerically.

5.4 Numerical policy improvement

As stated before, the complexity of the expression of the policy prevents us from applying more improvement steps explicitly. We therefore turn to numerical computations. As in Section 4.6, we need to truncate the state space to be able to solve for equilibrium distributions and the Howard equation.

As before, the necessary size of the state space depends on the load of the system. The closer to instability, the higher number of states there should be to avoid accumulation at the state space borders. In the following examples, we use $30^2 = 900$ states as a compromise between computation time and results' accuracy.

Low interference We choose a set of parameters that places us in the case $2\mu_1 > \mu_0$, and that comply with the stability conditions:

$$\mu_0 = 1.2, \mu_1 = 0.9, \lambda_1 = 0.2, \lambda_2 = 0.3.$$

The optimal static policy is to serve both queues at rate μ_1, according to the condition (5.2). Solving the Markov process gives an average total queue length of 0.785714 users.

The first iteration, expressed previously, gives an average of 0.681884 users. The second iteration leads already to the optimal policy, as shown in Figure 28, with an average number of users of 0.680562. Indeed, applying the iteration step again gives only the same policy, which indicates we reached convergence.

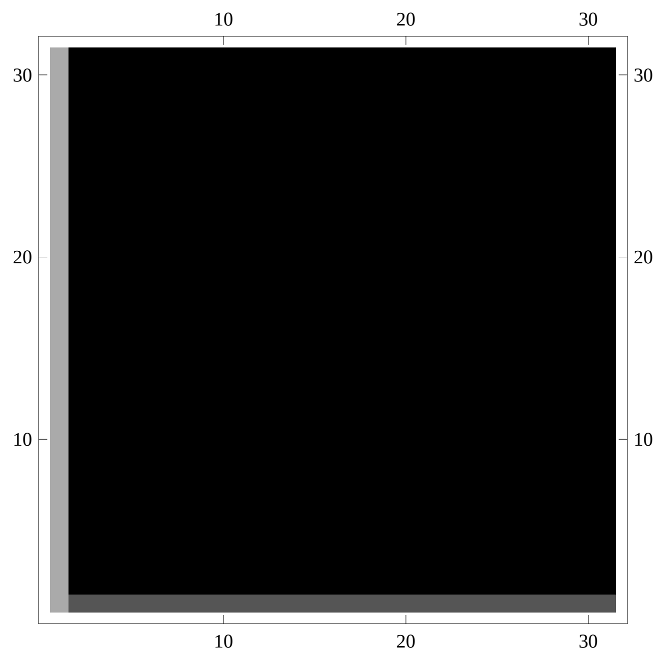

Figure 28: Policy iteration, low interference: serve both queues whenever possible, otherwise serve the non empty queue

High interference Now considering $2\mu_1 \leq \mu_0$, we experiment with the following parameters:

$$\mu_0 = 1.2, \mu_1 = 0.4, \lambda_1 = 0.3, \lambda_2 = 0.4.$$

The optimal static policy is to serve the queues at rates $(0.355051, 0.489898)$. This leads to an average of 9.89898 users in the system.

The first iteration is the simple switching curve described previously, and an average of users of 1.4. Although we know that this situation is already optimal, the algorithm does not stop yet. The average does not decrease anymore, but the policy oscillates through several seemingly random similar states, as shown in Figure 29.

Depending on the actual parameters, the algorithm can either stop in a few steps, loop back at some point to a previously found policy—in which case the algorithm will loop forever—, or go on for a possibly very large number of steps. However, in all the experiments done, the average stabilizes in a few steps only. For this reason, it was found more useful to set the stopping condition on the average (when it does not change by more than an arbitrary small value such as 10^{-8}) rather than on the policy itself.

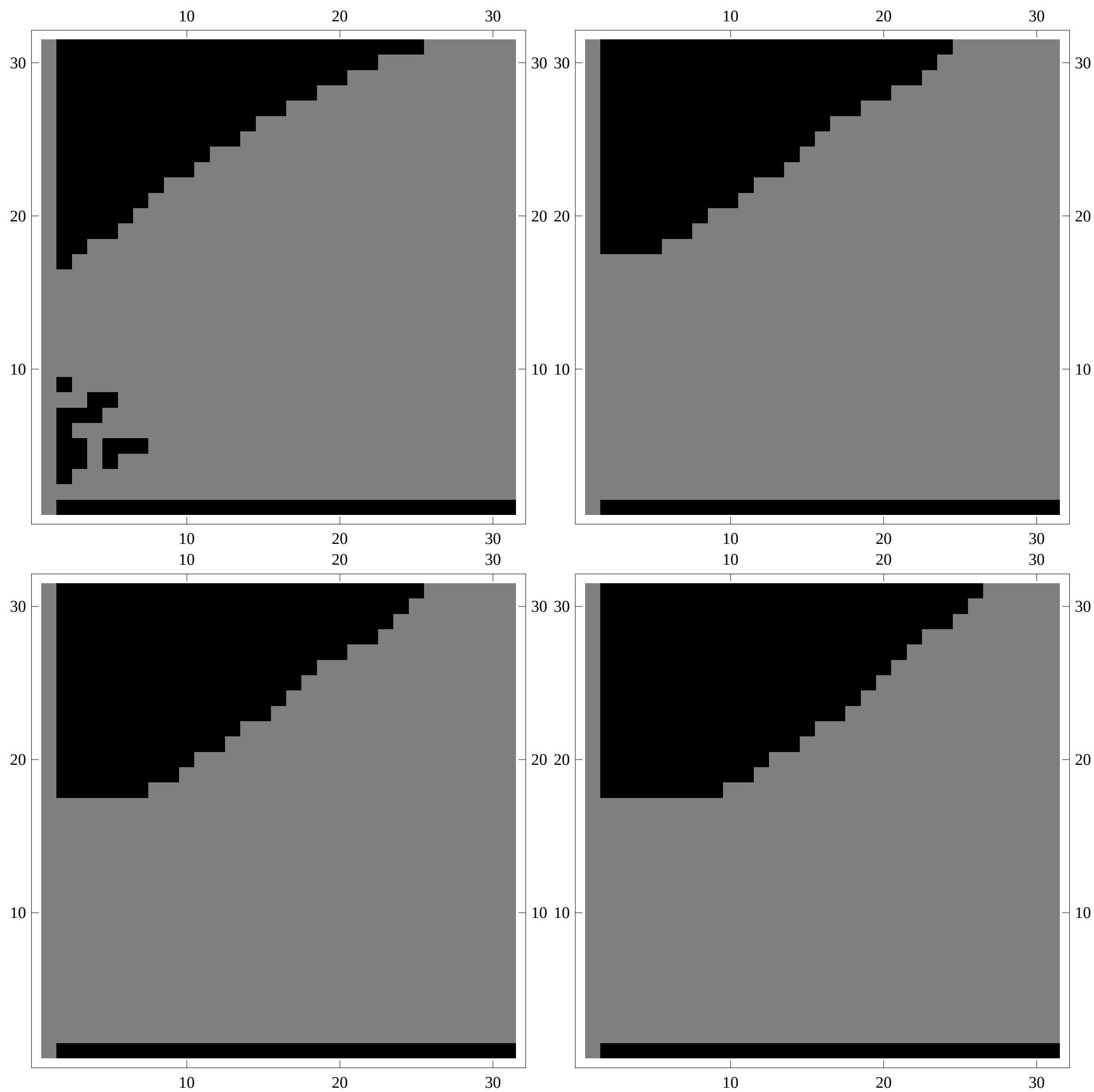

Figure 29: Policy iteration, high interference: always serve only one queue at a time

In any case, these considerations are mostly for the sake of understanding how the policy improvement algorithm works, since we know that the optimal policy is always reached after the first step, for high interference. All subsequent policies make indeed no use of mode "11" and are therefore stochastically optimal.

Notes on the numerical experiments

- Although we presented here only two examples, they happen to be quite representative of all the numerical experiments we made. In low interference, the optimal policy is often reached at step 2, rarely at step 3, never more, and the algorithm converges. In high interference, the optimal policy is—as proven before—reached at the first step, but it usually does not converge immediately, and the stopping condition has to be the average number of users.

- The policy improvement algorithm as described by Howard gives the average-optimal policy only. In this case it happens to be stochastically optimal as well, but this is not a general result.

- Running the algorithm in unstable cases gives rather incoherent results. Although the policies returned are indeed optimal, they are so only in the context of the truncated state spaces, which are then not representative of the infinite space we are studying. The patterns described before disappear. See for instance Figure 30 for parameters $\mu_0 = 1.2, \mu_1 = 0.8, \lambda_1 = 0.6, \lambda_2 = 0.9$.

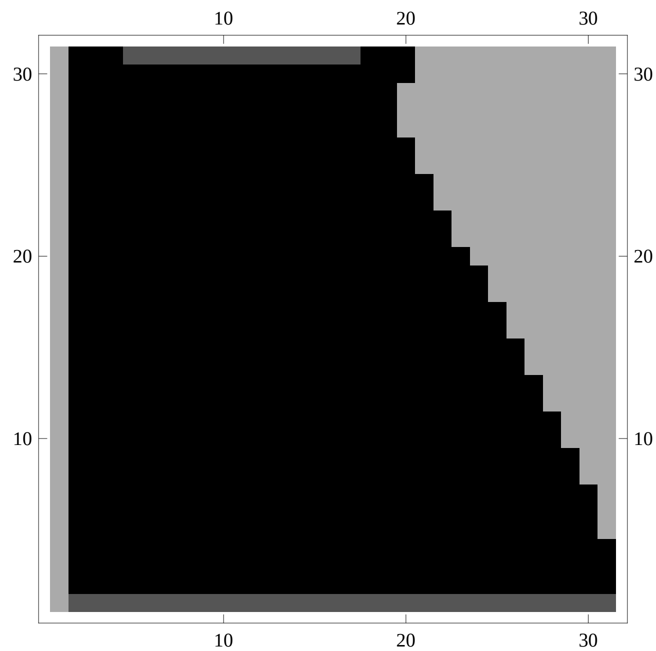

Figure 30: Optimal policy reached for an unstable case

6 Quantifying the improvement

We have now shown by two different means that the average number of users in the system (and therefore the delay) can be optimally reduced by using the following policy:

If one queue is empty, serve only the other one. If both queues are non empty

- in "low interference", always serve both queues at the same time.
- in "high interference", never serve both queues at the same time[3].

However, we have not yet compared this optimal policy $\tilde{\pi}$ with the basic policy π_0 we tried to improve in the first place (that is, to always keep both base stations on at all times). We will therefore compute for numerical examples the *gain*, defined as the ratio of the average numbers of users for the basic policy over the optimal policy:

$$G(\lambda_1, \lambda_2, \mu_0, \mu_1) = \frac{\bar{N}^{\pi_0}}{\bar{N}^{\tilde{\pi}}}.$$

Note that when using policy iteration to reach the optimal policy, the gain does not evolve much after the first step. This is a typical behavior of policy improvement: although the algorithm may take a several steps to converge, the average cost value that we try to optimize evolves very little after the first few steps. In cases where the optimal policy is unknown or difficult to describe (other than by actually waiting the convergence of the algorithm), a good approximation can usually be found in the first or second step.

Of course, since here the optimal policy is entirely known, the actual gain can be computed.

6.1 Gain

To obtain a general idea of the gain brought by the optimal policy, a few numerical examples are insufficient, hence the necessity of a systematic parameter space exploration. Of course, the latter being continuous and infinite, a few measures are to be taken to make this realistic.

To begin with, note that scaling all the parameters by a constant k leaves all results unchanged, for any policy. Indeed, replacing $\lambda_1, \lambda_2, \mu_0, \mu_1$ by $k\lambda_1, k\lambda_2, k\mu_0, k\mu_1$ in the Markov process' transition matrix $\mathbf{Q}$ simply multiplies it by k, because all transition rates are simple linear combinations of the parameters. And naturally, the nullspace of $k\mathbf{Q}$ is the same as the one of $\mathbf{Q}$, which means that the steady state distribution found by solving $\boldsymbol{\pi}^T\mathbf{Q} = \mathbf{0}$ is unchanged.

[3]The actual choice of which queue to serve depending on the state does not matter, except for fairness purposes, as seen in Section 7

We can therefore normalize every point in the 4-dimensional parameter space to a vector of three parameters, by setting for instance $\mu_0 = 1$:

$$(\lambda_1, \lambda_2, \mu_1, \mu_0) \to (\lambda_1/\mu_0, \lambda_2/\mu_0, \mu_1/\mu_0, 1).$$

Furthermore, as the system is symmetrical with respect to the two queues, we will compute the gain only for $\lambda_1 \leq \lambda_2$. Similarly, we know that for the optimal policy, when $\mu_1 < \mu_0/2 = 0.5$, the result is the same as when $\mu_1 = \mu_0/2 = 0.5$. A partial memoization technique is beneficial here.

Another concern which arose previously when solving for steady state distributions, is the effect of truncating the state space on the accuracy of the results. The closer to instability, the higher will be the probability of reaching the upper boundaries of the state space, which represent blocking states (user is denied access to the system and considered lost forever). While this is a perfectly valid model, it is not the one we studied, and the results in this situation would be far from the ones wanted.

We will therefore avoid computing the gain in unstable cases, and consider with precautions the results found close to stability region boundaries.

The same goes with the basic policy. Serving at all times at rate (μ_1, μ_1) makes the system two independent queues, and the stability condition is $\max(\lambda_1, \lambda_2) < \mu_1$. Computing a gain value when the basic system is unstable makes no sense: its average number of users is infinite.

Given the stability conditions described earlier for the optimal policy, we know that any load stable for π_0 is stable for $\tilde{\pi}$ (see Figure 31).

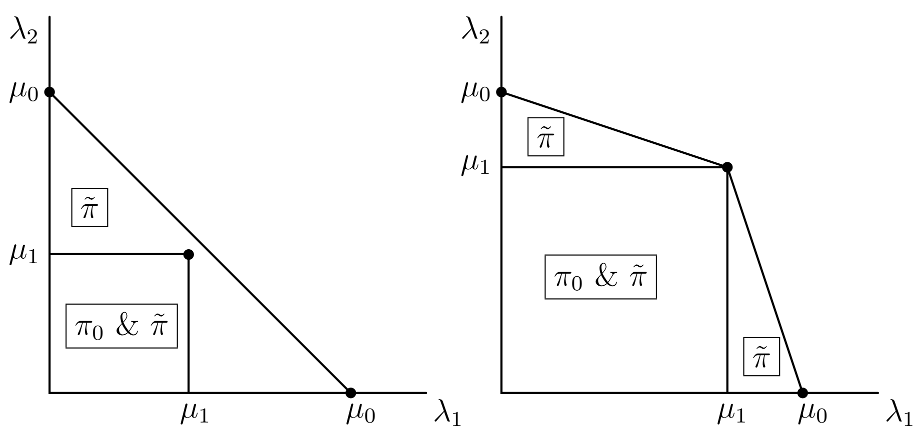

Figure 31: Stability of the basic and the optimal policies in high and low interference

We will therefore restrict the parameter space to the values that keep both the basic and the optimal policies stable. That is, λ_1 and λ_2 in the range $[0, \mu_1]$.

Finally, a sampling resolution δ is to be chosen. Keeping in mind that we have three parameters, the computation time will be bounded in $O(1/\delta^3)$. Actually, since the range of the λ_i varies according to μ_1, we chose a variable δ: instead of fixing the resolution, we fix the number of equidistant values for every parameter to some value N. Taking symmetry in account, the number of samples is finally close to $\frac{N^3}{2}$.

Figures 32 and 33 show the gain for two examples of μ_1. As foreseen, the gain tends to increase drastically near the stability limit, so we should instead concentrate on the lower loads. With this in mind, for $\mu_1 = 0.2$, that is in the high interference case, G ranges to about [5,10], and for $\mu_1 = 0.8$, to about [1.2,1.5].

These represent the general tendency of the results, and make perfect sense. When interference is high to begin with, interference mitigation can be very efficient. When interference was low anyway, there is less room for improvement. See Appendix C for complete tabulated results.

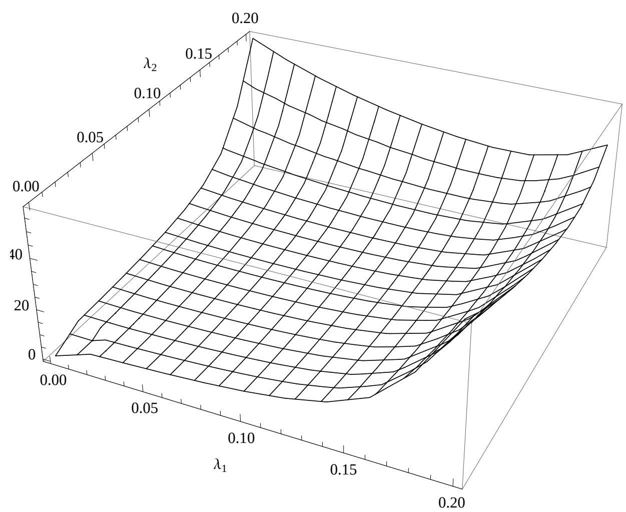

Figure 32: G for $\mu_1 = 0.2$

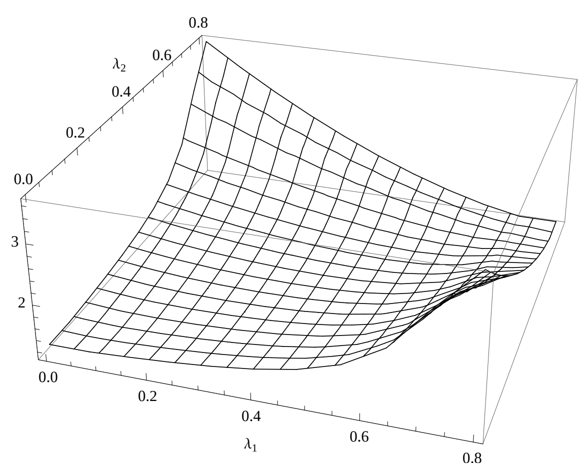

Figure 33: G for $\mu_1 = 0.8$

7 Fairness

So far, we only considered the optimization of the total number of users, including both queues without distinction. The low interference case led to a single optimal policy, for which we can attempt to measure fairness. The high interference led to a whole set of policies, meaning that we can optimize fairness as well, in the context of optimal policies with respect to the total number of users.

We will consider the system "fair" if the average delays in each queue $\bar{T}_1$ and $\bar{T}_2$ are close to each other. Ideally, a perfectly fair system would have $\bar{T}_1 = \bar{T}_2$. In order to measure the fairness, we introduce the expression $|\bar{T}_1 - \bar{T}_2|$. This could in fact be called "unfairness", since its value is zero for a perfectly fair system, and increases with the difference between the $\bar{T}_i$.

7.1 Low interference

From a truncated transition matrix representing the Markov process of the optimal policy, we compute a steady state distribution, which itself is used to compute the average numbers of users in each queue $\bar{N}_1$ and $\bar{N}_2$. From Little's formula comes $\bar{T}_i = \bar{N}_i/\lambda_i$. Figure 34 shows an example of $|\bar{T}_1 - \bar{T}_2|$ as a function of the arrival rates λ_1 and λ_2, for $\mu_1 = 0.8$.

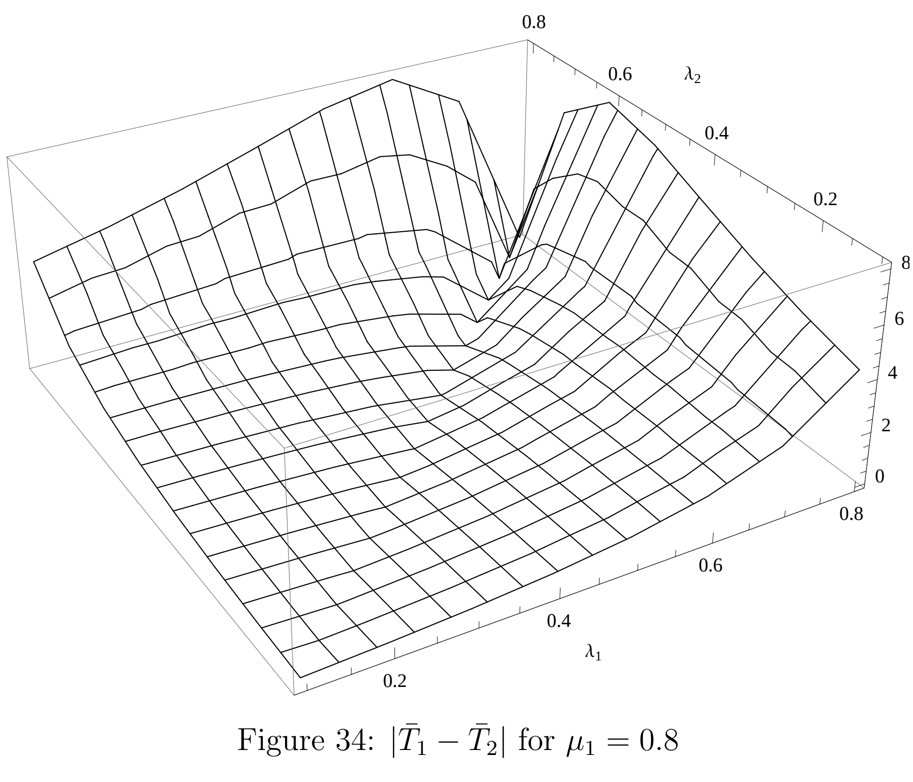

Figure 34: $|\bar{T}_1 - \bar{T}_2|$ for $\mu_1 = 0.8$

Without surprise, the functions $\bar{T}_1$ and $\bar{T}_2$ are symmetrical to one another with respect to the $\lambda_1 = \lambda_2$ line. Furthermore, for $\lambda_1 < \lambda_2$, $\bar{T}_1 < \bar{T}_2$.

We can safely generalize this example to all low interference cases. Indeed, since the two queues are served with the same rate most of the time, the one with the lowest load will have the lowest average delay. And since the policy is strictly identical for all low interference cases, this should be general as well.

In other words, the system is biased towards the queue with the lowest arrival rate, and is fair only if the two arrival rates are equal.

7.2 High interference

In high interference, there is still a degree of freedom. Indeed, we know that *any* policy that never serves both queues at the same time will be optimal for the total number of users. It may be possible to choose a policy that also optimizes fairness as a secondary objective.

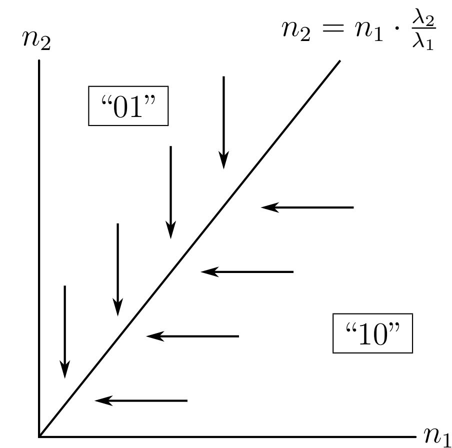

Figure 35: Heuristically fair optimal policy

Although no proof is given, the policy presented in Figure 35 seems to be a good candidate. It is a simple switching curve of equation

$$n_2 = n_1 \cdot \frac{\lambda_2}{\lambda_1}.$$

We can see heuristically how fair this policy can be: the switching curve acts as a state "attractor". Indeed, below the curve, only the queue 1 is served, which makes n_1 decrease. Similarly, above the curve, only the queue 2 is served, which makes n_2 decrease. Therefore, most of the time, the current state will tend to be close to the switching curve—as represented by the arrows on Figure 35—and we can assume

$$\bar{N}_2 \simeq \bar{N}_1 \cdot \frac{\lambda_2}{\lambda_1},$$

which gives an approximation of the fairness condition

$$\frac{\bar{N}_1}{\lambda_1} \simeq \frac{\bar{N}_2}{\lambda_2}$$
$$\bar{T}_1 \simeq \bar{T}_2.$$

Note that among policies determined by linear switching curves, this one is the closest to fairness (any other coefficient would lead to $\bar{T}_1 \not\approx \bar{T}_2$). Of course, it is possible that another type of switching curve, or even a policy not determined by a switching curve at all, is in fact more fair.

8 Dynamic analysis of some OFDMA reuse patterns

This section is largely based on [3], and originally constituted an introductory work to get acquainted with mathematical software.

Orthogonal Frequency-Division Multiple Access (OFDMA) allows for the division of the bandwidth into subbands, the widths and power levels of which can be selectively chosen for different classes of users, depending in particular on their position in the cell. This combined with careful network planning—in this case frequency reuse patterns—can significantly reduce inter-cell interference and thus increase the capacity of the network.

Frequency reuse schemes are usually evaluated in static scenarios (see references in [3]), which do not take into account the random nature of traffic: a fixed number of users are placed randomly in a network, and capacities are computed. However in real systems, users come and go randomly, making the number of active connections time dependent. We consider here a downlink network in which users request files of random sizes from random locations, at random time instants. The objective is to determine the *capacity* of a cell, defined as the maximum traffic that leads to a stable queuing system.

8.1 Reuse patterns

Frequency reuse consists of dividing the available bandwidth into a fixed number of subbands, and assigning them to cells or subcells geographically. The point is to increase the distance between areas that use the same bands, in order to reduce the interference that each one causes on the others.

We consider here "full", "hard", "soft" and "fractional" reuse. For each we provide an example in an hexagonal network. Note that the figures show only one tile, which would be repeated to form a bigger network. Each color represents one frequency band, and two colors side by side indicate that both bands are used in the given area.

For a given reuse pattern, we call $1/K$ the reuse factor, that is, the fraction of bandwidth used by each cell.

Full reuse This is the most basic assignment of frequencies, and does not actually reduce interference. Indeed, every cell uses the full frequency band, as shown in Figure 36.

Hard reuse The frequency band is divided into a fixed number of subbands, normally allocated to cells according to a regular pattern. Figure 37 shows for instance a reuse factor of $1/K = 1/3$, that is there are 3 subbands, and each cell therefore uses $1/3$ of the total bandwidth. More generally, with hard reuse, $1/K < 1$.

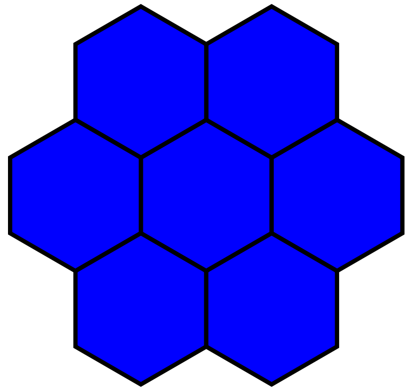

Figure 36: Full reuse, $1/K = 1$

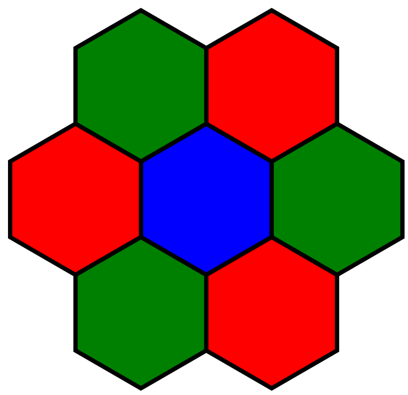

Figure 37: Hard reuse, $1/K = 1/3$

Soft reuse The frequency band is divided into a fixed number of equally large subbands. The users in the cell are divided between "near" users and "far" users, depending on their distance to the base station, or quite equivalently their SINR (Signal to Interference and Noise Ratio). The far users use a pattern similar to hard reuse, while the near users use the remaining bands. The reuse factor is equal to 1, since each cell uses every subband. Figure 38 shows an example with 3 subbands.

We see here two variable parameters appear: the near/far distance separation, and the near/far power ratio.

Fractional reuse The frequency band is divided into: one "near" band used by all near users, and the rest into any number of equally wide "far" subbands as a hard reuse scheme for the far users (Figure 39). Note that $1/K < 1$, hence the name "fractional" reuse.

There are now three parameters: the near/far distance and the near/far power ratio (as for soft reuse), plus the near/far bandwidth ratio.

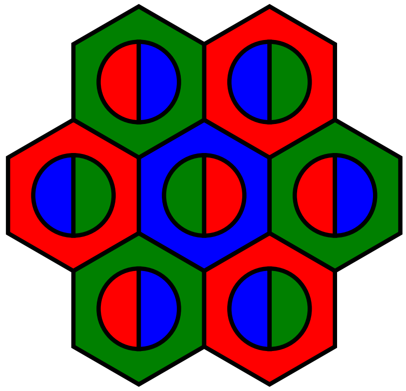

Figure 38: Soft reuse, 3 subbands, $1/K = 1$

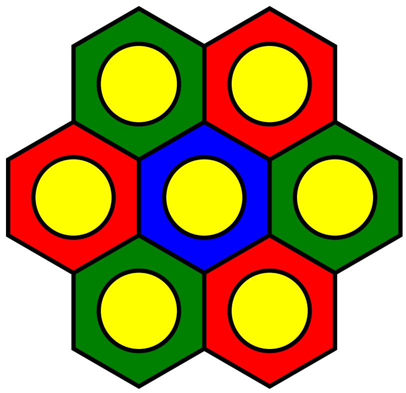

Figure 39: Fractional reuse, 3 "far" subbands, $1/K < 1$

8.2 Teletraffic model

For each class of users (that is, every subband used in a certain cell), denote $R(u)$ the throughput that would be achieved by a user whose position is $u \in \mathcal{C}$ in the cell, if there were no other users in his class. With fair sharing of the radio resource (in time and/or frequency), and $N(t)$ active users in the class at time t, this throughput becomes $R(u)/N(t)$.

Each active user is downloading a file at rate $R(u)/N(t)$. We assume that u is fixed for the duration of the transfer. However, the number of active users does vary, and therefore a flow of size s started at time t_0 will finish at a time t_1 such that

$$\int_{t_0}^{t_1} \frac{R(u)}{N(t)} \, \mathrm{d}t = s.$$

The rate of generation of flows (that is, of arrival of clients) being λ, and its geographic distribution $\delta(u)$, we know that new flows arrive at rate $\lambda\delta(u)\,\mathrm{d}u$ in an infinitesimal area around $u \in \mathcal{C}$. For the purpose of simplification, we assume that this flow density is uniform, and that $\mathcal{C}$ has a unit area. Therefore, $\delta(u) = 1, \quad \forall u \in \mathcal{C}$.

Flows are assumed to be independent and identically distributed, with average size σ. Therefore the average traffic intensity generated by the cell is $\lambda\sigma$ bit/s.

Markov process The $N(t)$-dimensional vector $X(t)$ of the positions of the active users of the cell form a Markov process. We have already seen that the arrival transition is done at rate

$$\int_{\mathcal{C}} \lambda\delta(u)\,\mathrm{d}u = \lambda.$$

The completion rate on the other hand, for a user at $u \in \mathcal{C}$ is equal to

$$\frac{R(u)}{N(t)\sigma}.$$

Averaged for $N(t)$ users distributed on $\mathcal{C}$ according to distribution $\delta(u)$, the overall average service time becomes (see [3]):

$$\int_{\mathcal{C}} \frac{\sigma}{R(u)}\delta(u)\,\mathrm{d}u.$$

We therefore have a load of

$$\rho = \lambda \int_{\mathcal{C}} \frac{\sigma}{R(u)}\delta(u)\,\mathrm{d}u.$$

Stability For the queue to be stable, the load must be less than one, which gives the condition

$$\lambda\sigma < \left(\int_{\mathcal{C}} \frac{\delta(u)}{R(u)}\,\mathrm{d}u\right)^{-1}.$$

We call this maximum sustainable traffic the "capacity" C of the cell for the given class of users.

Full and hard reuse In these reuse schemes, each cell has only one class of users. The capacity is thus directly

$$C = \left(\int_{\mathcal{C}} \frac{\delta(u)}{R(u)}\,\mathrm{d}u\right)^{-1}.$$

Soft and fractional reuse Denoting the "near" and "far" regions $\mathcal{C}_1$ and $\mathcal{C}_2$, each class is an independent queue and the previous result is applicable to each:

$$C_1 = \left(\int_{\mathcal{C}_1} \frac{\delta(u)}{R_1(u)}\,\mathrm{d}u\right)^{-1}, \quad C_2 = \left(\int_{\mathcal{C}_2} \frac{\delta(u)}{R_2(u)}\,\mathrm{d}u\right)^{-1}.$$

In order for the whole cell to constitute a stable system, both queues must be stable, and the overall capacity is set on the most constraining class:

$$C = \min(C_1, C_2).$$

The relative values of C_1 and C_2 depend on the partitioning of the cell into the regions $\mathcal{C}_1$ and $\mathcal{C}_2$. Intuitively, the capacity for one class is a strictly decreasing function of the area of the region: since λ is constant on the cell, a smaller sub-region means less arrivals, for the same service rate. As the sum of areas of the two regions is constant, C_1 and C_2 are one increasing and the other decreasing. Therefore their minimum C reaches its maximum for $C_1 = C_2$ and the corresponding regions $\mathcal{C}_1^*$ and $\mathcal{C}_2^*$.

8.3 Radio model

The radio channel is characterized by additive white Gaussian noise (AWGN) and therefore by Shannon's formula

$$R(u) = W \lg(1 + \eta(u)),$$

where W is the width of the considered band in Hz, and $\eta(u)$ the SINR at point u. The throughput $R(u)$ that we saw already is of course in bit/s. This channel coding being quite optimistic compared to actual technology (this is the maximum attainable rate, using arbitrarily complex codings), it will give an upper bound for the results on the capacity.

We ignore thermal noise and fast fading to concentrate on the interference only. Placing the cell's base station at the origin, and calling $\mathcal{B}$ the set of interfering base stations, the SINR is simply

$$\eta(u) = \frac{|u|^{-\alpha}}{\sum_{v \in \mathcal{B}} |u - v|^{-\alpha}},$$

with $\alpha = 3.5$. Since no noise is actually considered, this is actually a Signal to Interference Ratio (SIR).

With directional antennas, an angular path loss is added:

$$h(\theta) = \frac{|\theta|}{\theta_0} h_0 \text{ dB},$$

with $h_0 = 9$ dB the angular path loss at sector edge (that is, at θ_0 degrees).

8.4 Network topology

6 types of network are studied:

- Linear and hexagonal. In linear networks, we use 2 subbands, 3 for hexagonal.
- Unsectorized and sectorized, that use directional antennas. Linear cells have two sectors, hexagonal cells have three sectors.

- Regular and randomized (only for sectorized cells). The base stations are distributed uniformly in a disk of area $1/4$ around their original positions. Antennas in hexagonal cells also take a direction uniformly distributed in a cone of angle $\pi/3$ around their original direction.

8.5 Computing capacity gains

Given the complexity of the expression of C, no symbolic result can be expected. Unfortunately, even numerical computation is quite challenging. Indeed, it requires:

- Fixing arbitrarily the network size, and placing the base stations. A network too small is unrepresentative, since it underestimates interference, and a bigger one implies long computation times.

- Computing the power levels from each base station to each point of the central cell, in order to get a SIR and $R(u)$. See an example of SIR in an hexagonal network, for hard reuse, in Figure 41.

- Integrate over the cell. Although this is fine in linear networks, it so happens that mathematical software has difficulties integrating over odd shapes (hexagons, intersected with circles in the case of soft reuse). Fixed step, Monte Carlo or other methods for numerical integration are quite inefficient and slow.

Furthermore, in the soft and fractional cases, we have respectively 2 and 3 variable parameters, that we need to optimize. This requires deciding on a resolution and sampling, thus multiplying the number of integrations. The very large amount of computation forces to severely decrease the resolution, and therefore lose precision.

So that they take any kind of meaning, results for random networks have to be averaged over a large number of network instances, here 100. The time to compute grows by as much.

On the other hand, it is proved in [3] that the optimal fractional scheme (in terms of near/far bandwidth ratio) is never better than the soft reuse scheme for the same near/far distance and power ratio parameters. We therefore omitted this scheme in the computations.

Finally, we decided to fix the hexagonal network size to a total of 81 base stations, as shown in Figure 40. In full reuse, every base station interferes, while in hard reuse with reuse factor $1/3$, only the highlighted base stations do.

For the linear networks, computation is a lot less intensive, so the network size can be augmented until a reasonable convergence of the result is found. See for instance Figure 42 for the details of signal and interference in full and hard reuse in a seemingly infinite linear network.

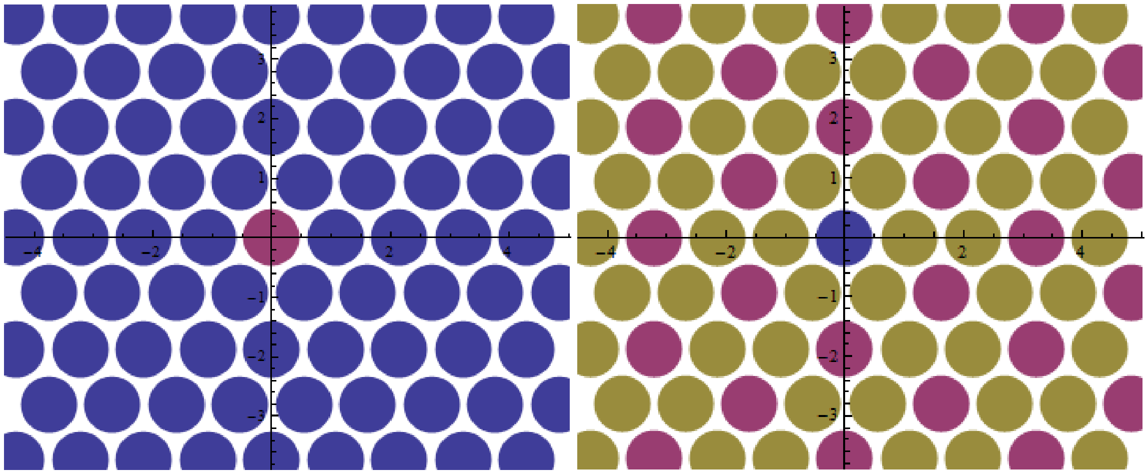

Figure 40: The considered hexagonal network, for full and hard reuse

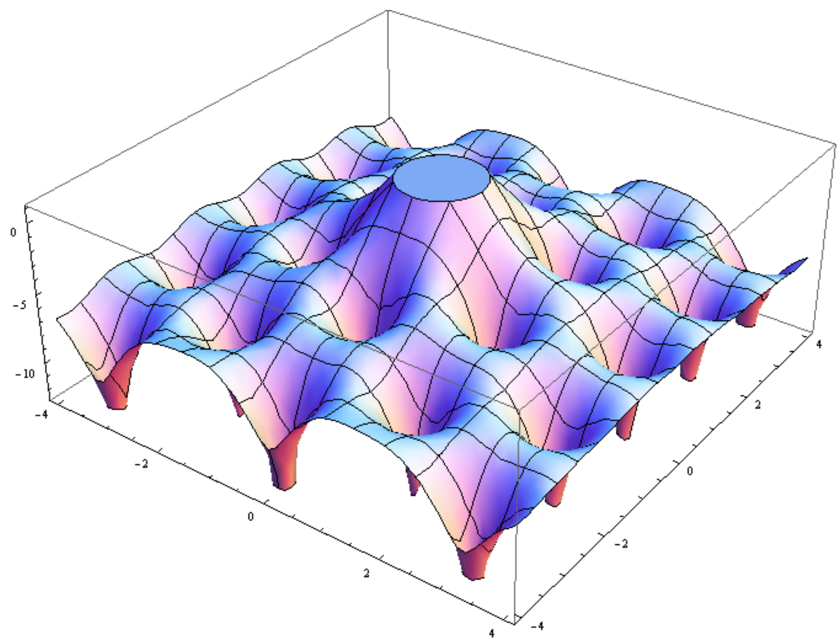

Figure 41: SIR in the considered hexagonal network, hard reuse

8.6 Results

For each network, the capacities in the hard and soft reuse are compared to the basic full reuse scheme. The resulting gain is given in the following tables.

Linear networks

Network	Hard reuse	Soft reuse
Unsectorized, regular	26%	53%
Sectorized, regular	17%	39%
Sectorized, random	16%	43%

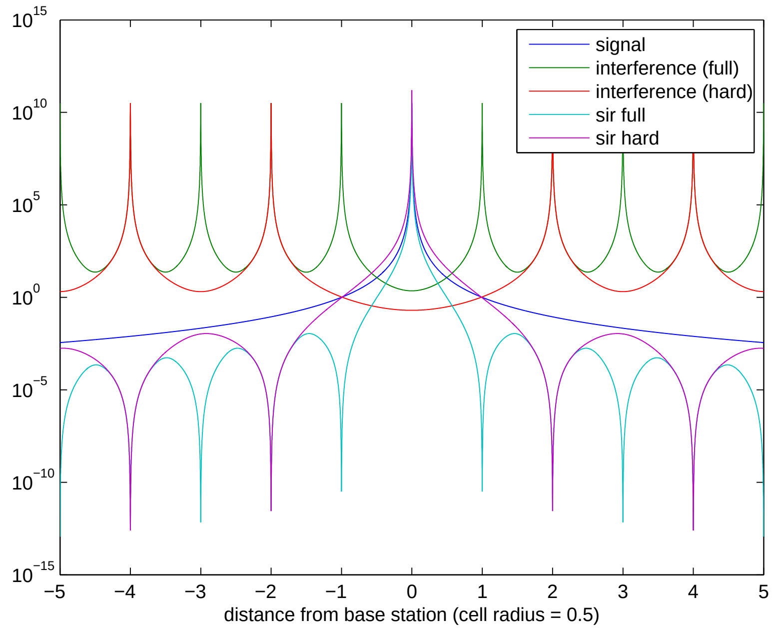

Figure 42: SIR in the (almost) infinite unsectorized linear network

Since computation is easy enough to allow big networks and convergence, these results should be fairly accurate. We note that reuse patterns indeed have a positive impact on the capacity. The random sectorized network, arguably close to reality, gives an improvement of 43% by applying soft reuse.

Reuse patterns are clearly beneficial to linear networks.

Hexagonal networks

Network	Hard reuse	Soft reuse
Unsectorized, regular	1%	20%
Sectorized, regular	-7%	3%

For hexagonal networks, on the other hand, the results should be considered inconclusive. Indeed, the sensitivity to network size makes it difficult to reach accurate numbers. Furthermore, computation times are almost prohibitive for one network, all the more for 100. Therefore, random networks—supposedly the closest to reality—have been omitted.

However, the tendency, while increasing network size, is to get poorer gains[4]. Furthermore, in the case of soft reuse, the gain is optimized with respect to near/far distance and near/far power ratio. It is reasonable to consider these numbers as upper bounds for the actual values. Assuming this, soft reuse seems to be beneficial

[4]The increasing number of interfering base stations have apparently a greater impact on hard and soft reuse than on full reuse.

only in unsectorized regular networks, with a maximum gain of 20%. If a real-life situation presents a rather regular network, it might be that soft reuse would improve its performance.

9 Summary

9.1 Accomplished work

We considered in this thesis the fact that two neighboring base stations in a cellular network interfere, thus reducing each other's signal to noise ratio and therefore the capacities of the cells. From a traffic point of view, this is translated to lower service rates, which lead to higher average number of users in the queues, and therefore higher average delays.

In order to improve the quality of service, we explored two ways of mitigating the interference between cells.

Time scheduling We first studied whether it was beneficial to allow base stations to be turned off part of the time. Indeed, intuitively, when a cell has few or no users, turning it off has little impact on its own delays, but the reduced interference can greatly improve its neighbors' condition.

We modeled the situation between two interfering base stations, and found that the results articulated around one condition on the parameters, splitting them into two cases. Denoting μ_0 the service rate of a station when the other one is off, and μ_1 its service rate when the other one is on and interferes, the cases depend on the relative order of the total service rate of the system when both stations are on, or only one:

- Low interference: $2\mu_1 > \mu_0$. Despite the interference, having both stations on leads to a higher service rate than having a single station on.

- High interference: $2\mu_1 \leq \mu_0$. Because of the interference, having both stations on leads to a lower service rate than having a single station on.

To define a set of policies to study, we considered that at each new event (arrival or departure) in the system, a mode should be chosen among the following three:

- Station 1 on, station 2 off

- Station 1 off, station 2 on

- Station 1 on, station 2 on

depending on the number of users in each cell n_1 and n_2.

Then by two different methods (first by direct proof, then by Markov Decision Process and numerical policy improvement) we found the optimal policy. That is, the policy that minimizes the average total number of users in the system[5]. The optimal policy depends on the degree of interference:

- Low interference: always keep both stations on, except when one queue is empty, in which case, serve only the other one.

[5] Actually, the policy is not only average-optimal, but stochastically optimal, as proven by the first method.

- High interference: never serve both queues at the same time, and if one queue is empty, serve the other one. Note that this does not define a unique policy, but a condition that indicates if a certain policy is optimal.

We then compared this optimal policy to the system in which no time scheduling is done—that is where both stations are always on—by computing numerically the ratio between the average number of users in each. This gain is of course highly dependent on all the system parameters, and a complete parameter space exploration was done. The general result is that

- in low interference, the order of magnitude of the improvement is between 1 and 2. The optimal policy is close to the original one, and there is little room for improvement at all.

- in high interference situations, on the other hand, the average number of users can be reduced by a factor up to 10.

An additional benefit of the optimal policy is that it greatly increases the stability region—in other words, the capacity of the cells—, especially in high interference. While some loads would make the original system unstable, they are supported by the time scheduled one.

Finally, we considered fairness in the system. While all the previous results concern the average *total* number of users, fairness is the measure of the difference of quality of service between the two cells. In low interference, there is a unique optimal policy, and fairness depends highly on the parameters: the cell with the lowest average delay is the one with the lowest arrival rate.

In high interference, on the other hand, a whole set of policies is optimal (regarding the total), and fairness can be accomplished as a secondary objective. The policy "serve cell 1 if $n_2 < n_1 \cdot \lambda_2/\lambda_1$, and vice versa"—where λ_i is the arrival rate in cell i—has been heuristically decided to be a good candidate.

Reuse patterns As an additional method for reducing inter-cell interference, we studied several frequency reuse patterns, that is, assignment of frequency subbands to geographical areas.

Although results were inconclusive for hexagonal cellular networks, linear networks can greatly benefit from soft reuse.

Soft reuse—in the case of linear networks—consists of dividing the band into two equally large subbands, and dividing each cell into "near" and "far" regions. Frequency subbands are then assigned alternatively so that a neighbor cell's "far" band is own's "near" band, and vice versa.

By carefully adjusting the separation between near and far users and the near/far power ratio, capacity gains up to 43% are achievable on randomized networks.

For hexagonal networks, the observed gain—even for undersized networks—were modest.

9.2 Future work

We present here a few suggestions for future work based on this one: ideas left unexplored, because they were out of the scope of this thesis, or by lack of time.

Asymmetrical service rates We assumed that both base stations had the same service rates (μ_0 when alone, μ_1 with interference), but different arrival rates (λ_i) to model the fact that the cells might have the same equipment, but be of different size or be located in areas with different population. The model could be extended to accommodate for different service rates in each cell.

This is basically the model studied in [2].

Number of cells A similar model could be applied to a network of three stations instead of two. The state space would become 3-dimensional, and the conditions on the system parameters might become quite intricate. A symbolic analysis may be impossible, but applying numerical policy improvement would be trivial (although limited by the state space size).

Combining reuse patterns and time scheduling In the present model, each cell has only one class of users, all sharing the same service rate. By applying a soft reuse pattern, each cell could have two classes of users (near and far), whose service rates depend on the surrounding cells' being on or off, and using the same subbands or not.

To model a linear network, where the reuse factor is 1/2, it would become 2 cells, with two sub cells each (near/far users), which makes 4 user classes, each with 1 arrival rate and 2 service rates. That is, 12 parameters, and a 4-dimensional state space. For a hexagonal network with 3 cells and 3 subbands, it becomes 6 user classes, each having 1 arrival rate and 4 service rates[6], which makes 30 parameters and a 6-dimensional state space. Simplifications (like symmetry) would probably be needed to get any result, even numerical, but the study might have interest.

Radio model Finally, no radio model was taken in account to actually determine the service rates μ_0 and μ_1. We separated every part of the study into "low" and "high" interference based on their relative values, but never determined if either one was realistic.

[6] Station alone, one interfering, the other interfering, both interfering.

References

[1] T. Bonald, S. Borst, and A. Proutière, "Inter-cell coordination in wireless data networks," *Euro. Trans. Telecoms. 2006*, vol. 17, pp. 303–312, 2006.

[2] I. Verloop and R. Núñez-Queija, "Asymptotically optimal parallel resource assignment with interference," *Queuing Systems*, vol. 65, pp. 43–92, 2010.

[3] T. Bonald and N. Hegde, "Capacity gains in some frequency reuse schemes in OFDMA networks," *IEEE Globecom 2009*, 2009.

[4] "Long Term Evolution (LTE): An Introduction – Ericsson White Paper," 2009.

[5] Motorola, "Long Term Evolution (LTE): A Technical Overview," 2008.

[6] R. Bellman, *Dynamic Programming.* Princeton University Press, 1957.

[7] R. Howard, *Dynamic Programming and Markov Processes.* Cambridge, Massachusetts: Technology Press-Wiley, 1960.

[8] R. Howard, "Semi-Markovian Decision Processes," *International Statistical Institute Bulletin*, 1962.

[9] S. Ross, *Applied probability models with optimization applications.* San Francisco: Holden-Day, 1970.

[10] J. Little, "A Proof of the Queueing Formula $L = \lambda W$," *Operations Research*, vol. 9, pp. 383–387, 1961.

[11] G. Fayolle and R. Iasnogorodski, "Two coupled processors: the reduction to a Riemann-Hilbert problem," *Z. Wahr. verw. Geb.*, vol. 47, pp. 325–351, 1979.

A Pyramid sequence

In Section 4.5 we compute the value of $V_k(x_1, x_2)$, which depends on the 5 neighboring states at the previous level $k-1$. The dependency thus propagates one state at a time until level $k = 0$ is reached, and at each level, the states that are concerned can be represented as a diamond shape, as shown in Figure A1.

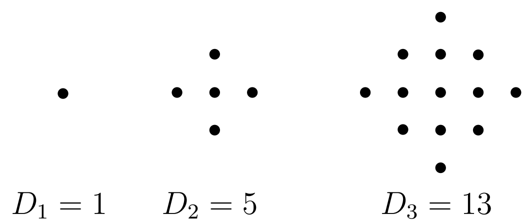

Figure A1: First terms of the "diamond" sequence

Calling D_k the number of attained states in level k, we get a "diamond" sequence. One can notice that to go from D_k to D_{k+1}, we add k states on each of the 4 sides of the diamond. This gives the simple recursive definition:

$$D_1 = 1$$
$$D_{k+1} = D_k + 4k.$$

It is then quite simple to derive an explicit expression:

$$D_k = 1 + 4\sum_{i=1}^{k-1} i$$
$$D_k = 1 + 4\frac{k(k-1)}{2}$$
$$D_k = 1 + 2k(k-1).$$

Then to get the desired result, let us call it P_k for "pyramid" sequence, we simply consider the series associated to the D_k sequence:

$$P_k = \sum_{i=1}^{k} D_i$$
$$P_k = \sum_{i=1}^{k} 1 + 2i(i-1)$$
$$P_k = k + 2\sum_{i=1}^{k} i^2 - 2\sum_{i=1}^{k} i$$
$$P_k = k + 2\frac{k(k+1)(2k+1)}{6} - 2\frac{k(k+1)}{2}$$
$$P_k = \frac{2k^3 + k}{3}.$$

B Minimum of exponentially distributed variables

Consider two independent exponentially distributed variables $P \sim \text{Exp}(p)$ and $Q \sim \text{Exp}(q)$, and their minimum $X = \min(P, Q)$. The problem is to determine the probability that each variable will actually be the minimum. In our case (Section 3.1), these are potential transition times, and the objective is to determine with what probability will a particular transition occur first. Without loss of generality, let us compute this probability for the variable P:

$$\begin{aligned}
P[X = P] &= P[\min(P, Q) = P] = P[P < Q] \\
&= \iint\limits_{\substack{x>0\\y>0}} \mathbf{1}_{x<y} P[(P, Q) = (x, y)] \,\mathrm{d}x \,\mathrm{d}y \\
&= \int\limits_{y=0}^{\infty} \int\limits_{x=0}^{y} p\exp(-px) \cdot q\exp(-qy) \,\mathrm{d}x \,\mathrm{d}y \\
&= \int\limits_{y=0}^{\infty} q\exp(-qy) \int\limits_{x=0}^{y} p\exp(-px) \,\mathrm{d}x \,\mathrm{d}y \\
&= \int\limits_{y=0}^{\infty} q\exp(-qy)(1 - \exp(-py)) \,\mathrm{d}y \\
&= \int\limits_{y=0}^{\infty} q\exp(-qy) \,\mathrm{d}y - q \int\limits_{y=0}^{\infty} \exp(-(p+q)y) \,\mathrm{d}y \\
&= 1 - \frac{q}{p+q} \\
&= \frac{p}{p+q}.
\end{aligned}$$

This result can be generalized. First note that for any number of independent exponentially distributed variables $X_i \sim \text{Exp}(q_i)$,

$$\min_i(X_i) \sim \text{Exp}(\Sigma_i q_i).$$

Indeed:

$$\begin{aligned}
P[\min(X_i) > x] &= P[X_1 > x, X_2 > x, \ldots] \\
&= \prod_i P[X_i > x] \\
&= \prod_i \exp(-q_i x) \\
&= \exp(-(\Sigma_i q_i)x).
\end{aligned}$$

Then, using the notation

$$X = \min_i(X_i), \quad X_i \sim \text{Exp}(q_i),$$

apply the previous result to X_j and $\min_{i \neq j}(X_i)$:

$$\begin{aligned}
P[X = X_j] &= P[\min_i(X_i) = X_j] = P[X_j < \min_{i \neq j}(X_i)] \\
&= \frac{q_j}{q_j + \sum_{i \neq j} q_i} \\
P[X = X_j] &= \frac{q_j}{\sum_i q_i}.
\end{aligned}$$

C Tabulated results for the gain of the optimal policy

Each table is for a fixed value of μ_1. The rows and columns are for regularly increasing values of λ_i, such that the first one is 0, and the last is equal to μ_1. For instance, for the first table, where $\mu_1 = 0.1$, the λ_i take the values

$$0, 0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09, 1$$

Note that the values for $\lambda_1 = \lambda_2 = 0$ make little sense and result from the ratio of two numbers that have no meaning. Indeed when the load is inexistent, the Markov Process is not irreducible, and solving for the equilibrium distribution by finding the nullspace of the transition matrix does not apply.

Furthermore, as discussed before, the closer to the stability limit, the less accurate and reliable the value prove to be.

$\mu_1 = 0.1$

	11.000	12.250	13.857	16.000	19.000	23.500	30.991	45.572	77.572	130.500
11.000	10.889	11.676	12.952	14.778	17.407	21.405	28.103	41.192	70.048	118.220
12.250	11.676	12.000	12.893	14.361	16.607	20.125	26.114	37.915	64.096	108.170
13.857	12.952	12.893	13.429	14.551	16.429	19.500	24.851	35.530	59.404	99.907
16.000	14.778	14.361	14.551	15.333	16.852	19.500	24.267	33.950	55.805	93.167
19.000	17.407	16.607	16.429	16.852	18.000	20.227	24.439	33.213	53.271	87.833
23.500	21.405	20.125	19.500	19.500	20.227	22.000	25.649	33.557	51.974	84.000
30.991	28.103	26.114	24.851	24.267	24.439	25.649	28.658	35.674	52.524	82.183
45.572	41.192	37.915	35.530	33.950	33.213	33.557	35.674	41.610	56.805	84.108
77.572	70.048	64.096	59.404	55.805	53.271	51.974	52.524	56.805	69.900	94.523
130.500	118.220	108.170	99.907	93.167	87.833	84.000	82.183	84.108	94.523	116.000

$\mu_1 = 0.2$

	5.444	6.000	6.714	7.667	9.000	11.000	14.329	20.805	34.950	58.000
5.444	5.333	5.657	6.206	7.000	8.148	9.897	12.830	18.559	31.132	51.803
6.000	5.657	5.750	6.107	6.722	7.679	9.187	11.765	16.851	28.087	46.708
6.714	6.206	6.107	6.286	6.728	7.500	8.786	11.045	15.569	25.652	42.489
7.667	7.000	6.722	6.728	7.000	7.593	8.667	10.634	14.660	23.733	39.000
9.000	8.148	7.679	7.500	7.593	8.000	8.864	10.553	14.125	22.299	36.167
11.000	9.897	9.187	8.786	8.667	8.864	9.500	10.908	14.047	21.401	34.000
14.329	12.830	11.765	11.045	10.634	10.553	10.908	11.997	14.689	21.260	32.675
20.805	18.559	16.851	15.569	14.660	14.125	14.047	14.689	16.842	22.585	32.823
34.950	31.132	28.087	25.652	23.733	22.299	21.401	21.260	22.585	27.278	36.175
58.000	51.803	46.708	42.489	39.000	36.167	34.000	32.675	32.823	36.175	43.500

$\mu_1 = 0.3$

	3.593	3.917	4.333	4.889	5.667	6.833	8.775	12.549	20.743	33.833
3.593	3.481	3.651	3.958	4.407	5.062	6.061	7.739	11.015	18.161	29.665
3.917	3.651	3.667	3.845	4.176	4.702	5.542	6.983	9.830	16.084	26.222
4.333	3.958	3.845	3.905	4.120	4.524	5.214	6.443	8.916	14.401	23.350
4.889	4.407	4.176	4.120	4.222	4.506	5.056	6.090	8.230	13.042	20.944
5.667	5.062	4.702	4.524	4.506	4.667	5.076	5.925	7.762	11.976	18.944
6.833	6.061	5.542	5.214	5.056	5.076	5.333	5.995	7.544	11.210	17.333
8.775	7.739	6.983	6.443	6.090	5.925	5.995	6.443	7.694	10.838	16.173
12.549	11.015	9.830	8.916	8.230	7.762	7.544	7.694	8.586	11.179	15.728
20.743	18.161	16.084	14.401	13.042	11.976	11.210	10.838	11.179	13.071	16.726
33.833	29.665	26.222	23.350	20.944	18.944	17.333	16.173	15.728	16.726	19.333

$\mu_1 = 0.4$

	2.667	2.875	3.143	3.500	4.000	4.750	5.998	8.421	13.639	21.750
2.667	2.556	2.648	2.833	3.111	3.519	4.143	5.193	7.243	11.675	18.596
2.875	2.648	2.625	2.714	2.903	3.214	3.719	4.591	6.319	10.083	15.979
3.143	2.833	2.714	2.714	2.816	3.036	3.429	4.142	5.589	8.776	13.780
3.500	3.111	2.903	2.816	2.833	2.963	3.250	3.817	5.015	7.697	11.917
4.000	3.519	3.214	3.036	2.963	3.000	3.182	3.610	4.581	6.814	10.333
4.750	4.143	3.719	3.429	3.250	3.182	3.250	3.538	4.292	6.115	9.000
5.998	5.193	4.591	4.142	3.817	3.610	3.538	3.666	4.197	5.628	7.922
8.421	7.243	6.319	5.589	5.015	4.581	4.292	4.197	4.458	5.476	7.183
13.639	11.675	10.083	8.776	7.697	6.814	6.115	5.628	5.475	5.969	7.013
21.750	18.596	15.979	13.780	11.917	10.333	9.000	7.922	7.182	7.011	7.295

$\mu_1 = 0.5$

	2.111	2.250	2.429	2.667	3.000	3.500	4.332	5.944	9.377	14.500
2.111	2.000	2.046	2.159	2.333	2.593	2.992	3.666	4.979	7.783	11.955
2.250	2.046	2.000	2.036	2.139	2.321	2.625	3.157	4.213	6.482	9.833
2.429	2.159	2.036	2.000	2.034	2.143	2.357	2.761	3.593	5.400	8.039
2.667	2.333	2.139	2.034	2.000	2.037	2.167	2.454	3.086	4.490	6.501
3.000	2.593	2.321	2.143	2.037	2.000	2.045	2.222	2.672	3.717	5.174
3.500	2.992	2.625	2.357	2.167	2.045	2.000	2.064	2.342	3.061	4.029
4.332	3.666	3.157	2.761	2.454	2.222	2.064	2.000	2.101	2.518	3.074
5.944	4.979	4.213	3.593	3.086	2.672	2.341	2.100	1.994	2.121	2.357
9.377	7.783	6.482	5.400	4.490	3.717	3.060	2.515	2.116	1.949	1.965
14.500	11.955	9.833	8.038	6.500	5.169	4.018	3.053	2.332	1.946	0.000

$\mu_1 = 0.6$

	1.741	1.833	1.952	2.111	2.333	2.667	3.221	4.293	6.535	9.667
1.741	1.667	1.698	1.775	1.895	2.073	2.347	2.809	3.706	5.584	8.180
1.833	1.698	1.667	1.692	1.764	1.892	2.105	2.477	3.215	4.767	6.879
1.952	1.775	1.692	1.667	1.691	1.768	1.921	2.210	2.802	4.059	5.732
2.111	1.895	1.764	1.691	1.667	1.693	1.787	1.995	2.452	3.442	4.715
2.333	2.073	1.892	1.768	1.693	1.667	1.700	1.829	2.159	2.903	3.812
2.667	2.347	2.105	1.921	1.787	1.700	1.667	1.714	1.918	2.432	3.015
3.221	2.809	2.477	2.210	1.995	1.829	1.714	1.666	1.740	2.031	2.329
4.293	3.706	3.215	2.802	2.452	2.159	1.918	1.740	1.653	1.712	1.769
6.535	5.584	4.767	4.059	3.442	2.903	2.432	2.031	1.712	1.507	1.369
9.667	8.180	6.879	5.732	4.715	3.812	3.015	2.329	1.769	1.369	0.000

$\mu_1 = 0.7$

	1.476	1.536	1.612	1.714	1.857	2.071	2.428	3.114	4.506	6.216
1.476	1.429	1.449	1.500	1.580	1.697	1.878	2.182	2.770	3.959	5.382
1.536	1.449	1.429	1.445	1.494	1.580	1.724	1.975	2.469	3.469	4.623
1.612	1.500	1.445	1.429	1.445	1.498	1.603	1.801	2.205	3.027	3.931
1.714	1.580	1.494	1.445	1.429	1.447	1.512	1.657	1.974	2.629	3.303
1.857	1.697	1.580	1.498	1.447	1.429	1.452	1.543	1.774	2.271	2.737
2.071	1.878	1.724	1.603	1.512	1.452	1.429	1.462	1.607	1.952	2.238
2.428	2.182	1.975	1.801	1.657	1.543	1.462	1.428	1.480	1.675	1.811
3.114	2.770	2.469	2.205	1.974	1.774	1.607	1.480	1.418	1.455	1.466
4.506	3.959	3.469	3.027	2.629	2.271	1.952	1.675	1.455	1.315	1.222
6.216	5.382	4.623	3.931	3.303	2.737	2.238	1.811	1.466	1.222	0.000

$\mu_1 = 0.8$

	1.278	1.312	1.357	1.417	1.500	1.625	1.833	2.229	2.985	3.659
1.278	1.250	1.262	1.293	1.341	1.411	1.519	1.700	2.047	2.702	3.248
1.312	1.262	1.250	1.260	1.290	1.343	1.430	1.583	1.880	2.438	2.866
1.357	1.293	1.260	1.250	1.260	1.293	1.358	1.481	1.728	2.192	2.514
1.417	1.341	1.290	1.260	1.250	1.262	1.303	1.394	1.591	1.964	2.193
1.500	1.411	1.343	1.293	1.262	1.250	1.265	1.323	1.468	1.754	1.904
1.625	1.519	1.430	1.358	1.303	1.265	1.250	1.271	1.364	1.564	1.650
1.833	1.700	1.583	1.481	1.394	1.323	1.271	1.250	1.283	1.396	1.431
2.229	2.047	1.880	1.728	1.591	1.468	1.364	1.283	1.242	1.262	1.252
2.985	2.702	2.438	2.192	1.964	1.754	1.564	1.396	1.262	1.177	1.121
3.659	3.248	2.866	2.514	2.193	1.904	1.650	1.431	1.252	1.121	0.000

$\mu_1 = 0.9$

	1.123	1.139	1.159	1.185	1.222	1.278	1.370	1.542	1.823	1.890
1.123	1.111	1.117	1.131	1.152	1.184	1.233	1.315	1.468	1.715	1.762
1.139	1.117	1.111	1.116	1.130	1.154	1.195	1.265	1.399	1.611	1.643
1.159	1.131	1.116	1.111	1.116	1.131	1.162	1.220	1.333	1.512	1.532
1.185	1.152	1.130	1.116	1.111	1.117	1.136	1.180	1.271	1.418	1.429
1.222	1.184	1.154	1.131	1.117	1.111	1.118	1.146	1.215	1.330	1.334
1.278	1.233	1.195	1.162	1.136	1.118	1.111	1.122	1.166	1.248	1.248
1.370	1.315	1.265	1.220	1.180	1.146	1.122	1.111	1.127	1.175	1.170
1.542	1.468	1.399	1.333	1.271	1.215	1.166	1.127	1.107	1.114	1.103
1.823	1.715	1.611	1.512	1.418	1.330	1.248	1.175	1.114	1.076	1.050
1.890	1.762	1.643	1.532	1.429	1.334	1.248	1.170	1.103	1.050	0.000

$\mu_1 = 1$

Obviously, when $\mu_1 = 1 = \mu_0$, the basic policy and the optimal policy are in fact identical. The values 0 at the border indicate instability.

	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.000
1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.000
1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.000
1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.000
1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.000
1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.000
1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.000
1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.000
1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.000
1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.000
0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000