Università degli Studi di Padova
Dipartimento di Ingegneria dell'Informazione
Corso di Laurea in Ingegneria delle Telecomunicazioni

Optimal Strategies for the Transmission of Compressed Biometric Time Series over Wireless Channels

Laureanda:
Maria Scalabrin

Relatore:
Michele Rossi

Correlatore:
Davide Zordan

Anno Accademico 2014/2015

a Emanuele

Abstract

With the pervasive advancement of mobile information and communication technologies, wearable devices are perceived as realms of increasing interest for researchers and application developers. Nevertheless, the limited size and batteries capacities of these devices can pose critical problems, thus minimizing a massive amount of information while maintaining the information losses negligible becomes an interesting option to reduce the power consumption and to extend the operational lifetime of these devices. As far as this scenario is concerned, it is suitable to deal with compressed biometric time series processing techniques. Data compression is achieved taking advantage of vector quantization, pattern matching, and codebook-based approaches, in the sense that the most recurrent data patterns can be identified within a biomedical signal and exploited as the input for the encoder at the transmitting side to generate, *at runtime*, a codebook based on the most representative among them. In particolar, the codebook indexes are associated to the code-vectors and transmitted to the decoder at the receiving side in place of the corresponding code-vectors, which are more resource demanding. Our model formulation focuses on the transmission of both the new code-vectors and the codebook indexes over a wireless unreliable channel described in terms of a non-negligible packet error probability, thus designing the optimal strategies for the scheduling of a Single-Server Multiple-Buffer queueing system, where a central controller is responsible for serving customers arriving in multiple buffers. In particular, the optimal strategies will be compared to the suboptimal strategies (also referred to as heuristic policies), which do not require full state information and turn out to be computationally light.

Contents

List of Figures

Chapter 1

Introduction

Figure 1.1: Bluetooth heart rate application for the ECG.

Wearable video cameras, smart glasses, smart watches, running GPS tracking devices, healthcare devices, gesture controllers are estimated to grow from 29 millions in 2014 to 172 millions in 2018, according to CCS Insight Predictions [1]. These devices can be exploited for entertainment and fitness applications, promoting and improving users learning and experience. These devices can also be exploited for medical purposes, allowing users to real-time monitor their health status in the comfort of their home and to transmit the medical results to a remote hospital structure.

With the pervasive advancement of mobile information and communication technologies, wearable devices are perceived as realms of increasing interest for researchers and application developers, promising to make gathering a massive amount of information more accessible. Nevertheless, the limited size and batteries capacities of these devices can pose critical problems, thus minimizing a massive amount of information while maintaining the information losses negligible becomes an interesting option to reduce the power consumption and to extend the operational lifetime of these devices.

As far as this scenario is concerned, it is suitable to deal with compressed biometric time series processing techniques, along with energy management and transfer paradigms in Wireless Sensor Networks (WSNs) and Energy Harvesting Wireless Sensor Networks (EHWSNs). A general overview on WSNs can be found in [2] [3] [4], where the nodes of these networks are provided with small sensor devices capable of interacting with the surrounding environment. A general overview on EHWSNs can be found in [5] [6] [7], where the nodes of these networks are equipped with onboard rechargeable batteries capable of scavenging power from the surrounding environment, thus leading to power-constrained issues.

A great deal of codebook-based approaches for the compression of biometric time series have been investigated throughout the last decades, with more attention to electrocardiogram (ECG) signals [8] [9] [10]. Less attention has been paid to electroencephalogram (EEG) signals, to phonocardiogram (PCG) signals, to photoplethysmogram (PPG) signals, and to other one dimensional physiological signals which exhibit recurrent data patterns [11] [12] [13]. The ECG is an important diagnostic test that checks the heart electrical and muscular functions, translating the heartbeat into a quasi periodic curve tracing and reporting heart conditions and disorders such as shortness of breath and palpitations. The processing and compression techniques for the ECG signals can be classified into three groups of techniques: the direct methods, the undirect methods, and the parametric methods. The direct methods process the biometric signals in the time domain, exploiting tools such as AZTEC, CORTES, and LTC. A comprehensive framework for these techniques can be found in [14] [15] [16]. The undirect methods process the bio-

metric signals in some transformed domains, exploiting tools such as FFT, DCT, and DWT. A comprehensive framework for these techniques can be found in [17] [18] [19]. The parametric methods process the biometric signals to extract some features and to predict the future behavior of the biometric signals themselves, taking advantage of vector quantization, pattern matching, and codebook-based approaches. The parametric methods also comprise Neural Networks [20] [21] [22].

This thesis relies on the basic idea that the most recurrent data patterns can be identified within a biomedical signal and exploited to generate, *at runtime*, a codebook based on the most representative among them. Also, codebook-based approaches enable relevant processing functions such as classification and learning [23] [24] [25], in the sense that the most recurrent data patterns can be exploited to assess the statistical properties of the biomedical signals and to adapt, *at runtime*, a codebook according to the statistical changes of the biomedical signals themselves. A general overview on codebook-based approaches is introduced next.

1.1 Codebook-based approaches

When the source output samples are correlated, sequences or blocks of source output samples tend to fall into output clusters. Encoding individual values provides a less efficient code than encoding sequences or blocks of values, thus a quantization method that works with individual source output samples (thus called scalar quantization) provides a less efficient code than a quantization method that works with sequences or blocks of source output samples (thus called vector quantization). In a codebook-based scenario, both the encoder and the decoder of a L-level vector quantizer share a common codebook of L ℓ-dimensional code-vectors, where the L ℓ-dimensional code-vectors are the quantized version of the ℓ-dimensional sequences or blocks of source output samples. Furthermore, the L ℓ-dimensional code-vectors have to be representative of the ℓ-dimensional sequences or blocks of source output samples, i.e., selecting the L-level vector quantizer output points to fall into output clusters gives a more accurate description of source output samples. In general, a

large set of ℓ-dimensional source output vectors is referred to as the training vectors set.

The code-vectors are associated to the codebook indexes. The current input vectors at the transmitting side are compared to the code-vectors. The best matched code-vectors are selected according to some error tolerance and the corresponding codebook indexes are transmitted to the decoder at the receiving side in place of the current input vectors, which are more resource demanding. Note that the codebook at the encoder must be synchronized with that at the decoder, i.e., the codebook at the transmitting side must be synchronized with that at the receiving side, otherwise the best matched code-vectors cannot be retrieved from the codebook indexes. If no code-vector that fullfills the constraints on the error tolerance is found, then a new code-vector must be added to both the codebook at the encoder and the codebook at the decoder, otherwise the transmitter and the receiver are said to be Out Of Synchronization (OOS), waiting for the new code-vector synchronization.

Note that the decoding process consists of a simple table lookup, thus taking advantage of vector quantization, pattern matching, and codebook-based approaches becomes an interesting option when the resources available for the decoding process are limited [26] [27] [28].

As far as the performance of the ℓ-level vector quantizer is concerned, it is common to deal with codebook design, codebook initialization, and fast codebook search algorithms [29] [30] [31]. The clustering approach is applied in the Lloyd Algorithm (LA) and in the Generalized Lloyd Algorithm (GLA), thus finding the optimum vector quantizer assuming that the statistical description of the source is known (in the LA) or unknown (in the GLA). The clustering approach works as follows: given the training set and the initial set of ℓ reproduction alphabet values, each element of the training set is assigned to the corresponding closest reproduction alphabet value, which is updated after computing the centroids of the elements of the training set assigned to it. At the end of the clustering approach there are ℓ quantizer partitions of sequences or blocks of source output samples clustered around the final set of ℓ reproduction alphabet values. In particular, the squared Euclidian distance measure is used

in the computation of the quantizer partitions in the LA, the mean-squared error distance measure is used in the computation of the quantizer partitions in the GLA, which is also referred to as the Linde-Buzo-Gray (LBG) algorithm. Note that there is no guarantee that the final solution of the LBG algorithm is the optimal solution. Furthermore, there is a relevant dependence of the final solution of the LBG algorithm on the codebook initialization. The Pairwise Nearest Neighbor (PNN) algorithm and the fast PNN algorithm promise to reduce computation and time requirements without sacrificing performance. The PNN algorithm works as follows: given the initial ℓ-dimensional codebook, the closest pair of elements of the ℓ-dimensional codebook are merged and replaced with their mean value, thus converting the ℓ-dimensional codebook into the optimal $(\ell - 1)$-dimensional codebook. The basic idea is to combine those clusters that would introduce the smallest increase in distortion when combined. Experimental results indicate that the fast PNN algorithm requires less than 5 percent of the amount of time needed for the LBG algorithm. The fast PNN algorithm allows for the suboptimal $log(\ell)$ nearest neighbor search algorithm in place of the optimal $O(\ell)$ full search algorithm. Some other fast codebook search algorithms are the Triangular Inequality Elimination (TIE) method, the Mean-distance-ordered Partial codebook Search (MPS) method, and the Double Test Algorithm (DTA) method [32] [33] [34].

1.2 Thesis outline

This thesis is organized as follows. Chapter 2 discusses our model formulation dealing with the application of the Dynamic Programming (DP) and Optimal Control theory, along with the setting of a Markov Decision Chain (also known as a discrete-time Markov Decision Process). Chapter 3 discusses the performance evaluation of the scheduling of a Single-Server Multiple-Buffer queueing system. In particular, Chapter 3 introduces the Value Iteration Algorithm. Chapter 4 concludes this work presenting possible future developments.

Chapter 2

Model formulation

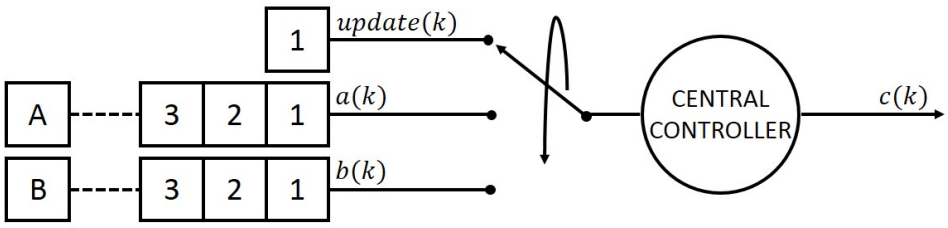

Figure 2.1: Single-Server Multiple-Buffer queueing system.

We consider a transmitter and a receiver communicating over a wireless unreliable channel described in terms of a non-negligible packet error probability. In particular, our scenario takes into account a wireless multipath Rayleigh fading channel, which will be well-characterized in terms of the received Signal to Noise Ratio (SNR) space according to the Markov Chain theoretical framework. Furthermore, our scenario takes into account the unidirectional communication from the transmitting side to the receiving side of compressed biometric time series coming from the application. Data compression is achieved tak-

ing advantage of vector quantization, pattern matching, and codebook-based approaches, in the sense that the most recurrent data patterns can be identified within a biomedical signal and exploited as the input for the encoder at the transmitting side to generate, *at runtime*, a codebook based on the most representative among them. In particolar, the codebook indexes are associated to the code-vectors and transmitted to the decoder at the receiving side in place of the corresponding code-vectors, which are more resource demanding. Furthermore, the encoder and the decoder need to share the same codebook, otherwise the transmitter and the receiver are said to be Out of Synchronization (OOS), waiting for the new code-vector synchronization. In this case, communication can damage the application, since the codebook indexes at the transmitting side and at the receiving side are associated to different pieces of information within a biomedical signal. As a consequence, the encoder and the decoder need to keep the codebook updated.

Our model formulation focuses on the transmission of both the new code-vectors and the codebook indexes over a wireless unreliable channel described in terms of a non-negligible packet error probability, thus designing the optimal strategies for the scheduling of a Single-Server Multiple-Buffer queueing system, where a central controller is responsible for serving customers arriving in multiple buffers. In particular, the optimal strategies will be compared to the suboptimal strategies (also referred to as heuristic policies), which do not require full state information and turn out to be computationally light.

According to the codebook-based approaches, we assume that the length of the new code-vectors is n times greater than the length of the codebook indexes, thus the transmission of the new code-vectors is more resource demanding than the transmission of the codebook indexes, as expected. If $n = 1$, then we suppose that the length of the new code-vectors is equal to the length of the codebook indexes. Furthermore, we assume that n time units are required for the transmission of the new code-vectors, otherwise just 1 time unit is required for the transmission of the codebook indexes.

Note that the scheduling of a Single-Server Multiple-Buffer queueing system turns out to be the scheduling of a Single-Server Multiple-Class queueing sys-

tem, where a central controller is responsible for serving customers arriving in the same buffer but belonging to multiple classes. The decision of which buffer to serve at each decision epoch in a Single-Server Multiple-Buffer queueing system becomes then the decision of which class to serve at each decision epoch in a Single-Server Multiple-Class queueing system.

The parallel queueing system takes into account two classes: the High Piority Queue (HPQ) and the Low Priority Queue (LPQ). HPQ provides the new code-vectors that must be added to both the codebook at the encoder and the codebook at the decoder, thus keeping it updated. LPQ provides the current codebook indexes that are transmitted to the decoder in place of the current input vectors, thus being responsible for throughput. Note that LPQ also contains the OOS codebook indexes that are transmitted to the decoder whenever the transmitter and the receiver are said to be OOS, waiting for the new code-vector synchronization. Thus, in the considered parallel queueing system we prefer to split LPQ into two further classes, referred to as LPQ_a and LPQ_b, to respectively distinguish between the current codebook indexes (*good codebook indexes*) and the OOS codebook indexes (*bad codebook indexes*). In particular, the OOS codebook indexes can damage the application, since the original pieces of information within a biomedical signal cannot be retrieved from the OOS codebook indexes.

The problem of designing the optimal strategies for the scheduling of a Single-Server Multiple-Buffer queueing system results in the application of the Dynamic Programming (DP) and Optimal Control theory, along with the setting of a Markov Decision Chain (also known as a discrete-time Markov Decision Process).

2.1 The DP and Optimal Control theory

The basic idea of the DP and Optimal Control theory [35] [36] [37] is to minimize the mathematical function that describes the undesirable outcome as the system evolves along the stochastic trajectory. From now on, this mathematical function is referred to as the cost function.

Our scenario takes into account a slotted time system where $k = 0, 1, 2, \ldots$ is the discrete-time index of the k-th decision epoch. The basic model formulation of the finite-state discrete-time DP problem captures the system evolution along the stochastic trajectory. The optimization is taken over the control or decision variable to be selected at index k according to the knowledge of both the state at index k and the past history of the whole process. In particular, the control or decision variable to be selected at index k is constrained to take values in a given non-empty subset that depends on both the state at index k and the past history of the whole process.

According to the theoretical framework considered in [35], the basic model formulation of the finite-state discrete-time DP problem is defined as

$$x_{k+1} = f(x_k, u_k, w_k) \quad k = 0, 1, 2, \ldots, N-1, \tag{2.1}$$

where $x_k \in \mathcal{X}$ is the state at index k, $u_k \in \mathcal{U}(x_k) \subset \mathcal{U}$ is the control or decision variable to be selected at index k, $w_k \in \mathcal{W}$ is the random parameter at index k (also called disturbance or noise depending on the context of the problem), f is the function that captures the system evolution along the stochastic trajectory, and N is the time horizon.

The cost function is accumulated over time from index $k = 0$ to index $k = N-1$ and depends on the random parameter at index k (also called disturbance or noise depending on the context of the problem), thus the basic model formulation of the finite-state discrete-time DP problem results in the minimization of the expected total cost, where the expectation is taken with respect to the joint distribution of the random variables involved.

An admissible policy $\pi = \{\mu_0, \mu_1, \ldots, \mu_{N-1}\} \in \Pi$ is a sequence of functions μ_k that map each state $x_k \in \mathcal{X}$ into the control or decision variable $u_k \in \mathcal{U}(x_k) \subset \mathcal{U}$, such that $u_k = \mu_k(x_k)$ $\forall x_k \in \mathcal{X}$ and $\forall k$.

Given the initial state $x_0 \in \mathcal{X}$ and the admissible policy $\pi = \{\mu_0, \mu_1, \ldots, \mu_{N-1}\} \in \Pi$, the expected total cost of the finite time horizon problem is defined as

$$J_{\pi,N}(x_0) = E\left\{ g(x_N) + \sum_{k=0}^{N-1} g(x_k, u_k, w_k) | x_0 \right\}, \tag{2.2}$$

where $g(x_k, u_k, w_k)$ is the cost per time unit at state $x_k \in \mathcal{X}$ when the control or decision variable $u_k \in \mathcal{U}(x_k) \subset \mathcal{U}$ is applied. Furthermore, $g(x_N)$ is the terminal cost. The DP and Optimal Control theory aims at minimizing the expected total cost of equation (2.2), thus finding the optimal expected total cost $J^*_{\pi,N}(x_0)$ and the optimal policy $\pi^* = \{\mu^*_0, \mu^*_1, \ldots, \mu^*_{N-1}\} \in \Pi$ of the finite time horizon problem, i.e.,

$$J^*_{\pi,N}(x_0) = \min_{\pi \in \Pi} J_{\pi,N}(x_0) \quad \text{and} \quad \pi^* = \arg\min_{\pi \in \Pi} J_{\pi,N}(x_0). \tag{2.3}$$

Note that, in general, the optimal expected total cost $J^*_{\pi,N}(x_0)$ and the optimal policy $\pi^* = \{\mu^*_0, \mu^*_1, \ldots, \mu^*_{N-1}\} \in \Pi$ of the finite time horizon problem are associated to the initial state $x_0 \in \mathcal{X}$.

The DP Algorithm and the Principle of Optimality due to Bellman are here introduced to characterize the considered scenario from a theoretical perspective.

DP Algorithm. Given the initial state $x_0 \in \mathcal{X}$ and the admissible policy $\pi = \{\mu_0, \mu_1, \ldots, \mu_{N-1}\} \in \Pi$, the optimal expected total cost for the basic model formulation of the finite-state discrete-time DP problem is equal to the cost-to-go $J_N(x_0)$ computed at the last step of the DP Algorithm, which starts from $J_0(x_N) = g(x_N)$ and proceeds backward in time from index $k = 0$ to index $k = N - 1$, i.e.,

$$J_{k+1}(x_{N-(k+1)}) = \min_{\substack{u_{N-(k+1)} \\ \in \mathcal{U}(x_{N-(k+1)}) \subset \mathcal{U}}} E\Big\{ g(x_{N-(k+1)}, u_{N-(k+1)}, w_{N-(k+1)}) + J_k(x_{N-k}) \Big\}. \tag{2.4}$$

Furthermore, if the admissible policy $\pi = \{\mu_0, \mu_1, \ldots, \mu_{N-1}\} \in \Pi$ attains the minimum in the right-hand side of equation (2.4), then the admissible policy $\pi = \{\mu_0, \mu_1, \ldots, \mu_{N-1}\} \in \Pi$ is the optimal policy.

The juxtification of the DP Algorithm is based on the Principle of Optimality.

Principle of Optimality. Let $\pi^* = \{\mu^*_0, \mu^*_1, \ldots, \mu^*_{N-1}\} \in \Pi$ be the optimal policy for the basic model formulation of the finite-state discrete-time DP problem. Let $\pi^*(i) = \{\mu^*_i, \mu^*_{i+1}, \ldots, \mu^*_{N-1}\}$ be the truncated version of the

optimal policy for the tail subproblem formulation of the finite-state discrete-time DP problem. Assume that the process is in state $x_i \in \mathcal{X}$ at index $k = i$ and aims at minimizing the cost-to-go from index $k = i$ to index $k = N$, i.e.,

$$J_{\pi,N}(x_i) = E\Big\{ g(x_N) + \sum_{k=i}^{N-1} g(x_k, u_k, w_k) | x_i \Big\}. \tag{2.5}$$

Then the truncated version of the optimal policy $\pi^*(i) = \{\mu_i^*, \mu_{i+1}^*, \ldots, \mu_{N-1}^*\}$ is the optimal policy for the tail subproblem formulation of the finite-state discrete-time DP problem.

The justification of the Principle of Optimality is quite intuitive. If $\pi^*(i) = \{\mu_i^*, \mu_{i+1}^*, \ldots, \mu_{N-1}^*\}$ were not the optimal policy for the tail subproblem formulation, we would be able to switch to the optimal policy for the tail subproblem formulation once the process is in state $x_i \in \mathcal{X}$ at index $k = i$, thus minimizing the cost-to-go from index $k = i$ to index $k = N$. The Principle of Optimality suggests that the optimal policy for the basic model formulation of the finite-state discrete-time DP problem is constructed backward in time, first defining the optimal policy for the tail subproblem formulation involving the last stage, then defining the optimal policy for the tail subproblem formulation involving the last two stages, then defining the optimal policy for the tail subproblem formulation involving the last three stages, etc.

2.1.1 The infinite time horizon problem

In this subsection, we investigate the finite-state discrete-time DP problem with infinite time horizon.

Given the initial state $x_0 \in \mathcal{X}$ and the admissible policy $\pi = \{\mu_0, \mu_1, \ldots\} \in \Pi$, the expected total cost of the infinite time horizon problem is defined as

$$J_{\pi,\infty}(x_0) = \lim_{N\to\infty} E\Big\{ \sum_{k=0}^{N-1} g(x_k, u_k, w_k) | x_0 \Big\}, \tag{2.6}$$

where $g(x_k, u_k, w_k)$ is the cost per time unit at state $x_k \in \mathcal{X}$ when the control or decision variable $u_k \in \mathcal{U}(x_k) \subset \mathcal{U}$ is applied. The DP and Optimal Control theory aims at minimizing the expected total cost of equation (2.6),

thus finding the optimal expected total cost $J^*_{\pi,\infty}(x_0)$ and the optimal policy $\pi^* = \{\mu^*_0, \mu^*_1 \dots\}$ of the infinite time horizon problem, i.e.,

$$J^*_{\pi,\infty}(x_0) = \min_{\pi \in \Pi} J_{\pi,\infty}(x_0) \quad \text{and} \quad \pi^* = \arg\min_{\pi \in \Pi} J_{\pi,\infty}(x_0). \tag{2.7}$$

Note that, in general, the optimal expected total cost $J^*_{\pi,\infty}(x_0)$ and the optimal policy $\pi^* = \{\mu^*_0, \mu^*_1, \dots\}$ of the infinite time horizon problem are associated to the initial state $x_0 \in \mathcal{X}$.

The assumption of an infinite number of stages is never satisfied in practice, but can be considered a reasonable approximation when dealing with problems involving a finite but large number of stages, a random number of stages, an unknown number of stages, a final instant which is distant in the future or subject to the control or decision variable to be selected at a final instant which is distant in the future. In particular, the difference in performance between the infinite time horizon problem and the corresponding finite time horizon problem becomes negligible as N tends to infinity.

There are several theoretical and computational issues regarding the relationship between the optimal expected total cost $J^*_{\pi,\infty}(x_0)$ of the infinite time horizon problem and the optimal expected total cost $J^*_{\pi,N}(x_0)$ of the corresponding finite time horizon problem. As far as the theoretical issues are concerned, the direct generalization of the limiting form of the DP Algorithm of equation (2.4) is known as the Bellman's equation, which is here introduced. As far as the computational issues are concerned, the direct generalization of the limiting form of the DP Algorithm of equation (2.4) is known as the Value Iteration Algorithm, which will be well-characterized in Chapter 3.

Bellman's equation. The optimal cost-to-go satisfies the functional equation

$$J^*(x) = \min_{u \in \mathcal{U}(x) \subset \mathcal{U}} E\Big\{ g(x,u,w) + J^*(f(x,u,w)) \Big\} \quad \forall x \in \mathcal{X}. \tag{2.8}$$

Furthermore, if the admissible policy $\pi = \{\mu, \mu, \dots\} \in \Pi$ attains the minimum in the right-hand side of equation (2.8), then the admissible policy $\pi = \{\mu, \mu, \dots\} \in \Pi$ is the optimal policy.

The expected total cost of equation (2.6) is the limit of the expected total cost of the corresponding finite time horizon problem, thus the optimal expected to-

tal cost of the infinite time horizon problem is the limit of the optimal expected total cost of the corresponding finite time horizon problem, i.e.,

$$J^*_{\pi,\infty}(x_0) = \lim_{N\to\infty} J^*_{\pi,N}(x_0) \quad \forall x_0 \in \mathcal{X}. \tag{2.9}$$

Note that there are some unusual exceptions when dealing with the convergence of equation (2.9) with unbounded $g(x_k, u_k, w_k)$, thus the infinite time horizon problem must be approached with some attention.

2.1.2 The average cost problem

The expected total cost of equation (2.6) can be unbounded as N tends to infinity, thus it is convenient to introduce the average cost problem.

Given the initial state $x_0 \in \mathcal{X}$ and the admissible policy $\pi = \{\mu_0, \mu_1, \dots\} \in \Pi$, the average cost of the infinite time horizon problem is defined as

$$J_\pi(x_0) = \lim_{N\to\infty} \frac{1}{E\{t_N\}} E\left\{ \sum_{k=0}^{N} g(x_k, u_k, w_k) | x_0 \right\}, \tag{2.10}$$

where $g(x_k, u_k, w_k)$ is the cost per time unit at state $x_k \in \mathcal{X}$ when the control or decision variable $u_k \in \mathcal{U}(x_k) \subset \mathcal{U}$ is applied. Furthermore, $E\{t_N\}$ is the expected completion time of the N-th decision epoch. The DP and Optimal Control theory aims at minimizing the average cost of equation (2.10), thus finding the optimal average cost $J^*_\pi(x_0)$ and the optimal policy $\pi^* = \{\mu^*_0, \mu^*_1, \dots\}$ of the infinite time horizon problem, i.e.,

$$J^*_\pi(x_0) = \min_{\pi\in\Pi} J_\pi(x_0) \quad \text{and} \quad \pi^* = \arg\min_{\pi\in\Pi} J_\pi(x_0). \tag{2.11}$$

Note that, in general, the optimal average cost $J^*_\pi(x_0)$ and the optimal policy $\pi^* = \{\mu^*_0, \mu^*_1, \dots\}$ of the infinite time horizon problem are associated to the initial state $x_0 \in \mathcal{X}$.

2.1.3 The semi-Markov problem

We have considered so far situations where $g(x_k, u_k, w_k)$ is the cost per time unit at state $x_k \in \mathcal{X}$ when the control or decision variable $u_k \in \mathcal{U}(x_k) \subset \mathcal{U}$ is

applied. In particular, we have considered so far situations where $g(x_k, u_k, w_k)$ does not depend on the time occurrence of the k-th decision epoch and on the transition time of the k-th decision epoch. In this subsection, we investigate the finite-state discrete-time DP problem where the time occurrence of the k-th decision epoch and the transition time of the k-th decision epoch depend on the state at index k and on the control or decision variable to be selected at index k. In other words, we take into account the semi-Markov problem in the discrete-time domain, where we include the information of time elapsed since the time occurrence of the k-th decision epoch as part of the evolution of the whole process. Furthermore, we juxtapose the definition of stage to the definition of time unit, thus the definition of expected cost per stage to the definition of expected cost per time unit.

Let τ be the transition time in the discrete-time domain. In particular, τ describes the time elapsed since the time occurrence of the k-th decision epoch.

Let t_k be the time occurrence of the k-th decision epoch in the discrete-time domain. In particular, t describes the continuous-time counterpart of t_k.

$x_k \in \mathcal{X}$ is the state at index k in the discrete-time domain. In particular, $x_k = x(t_k)$, i.e., $x(t) = x_k$ for $t_k \leq t < t_{k+1}$, where $x(t)$ describes the continuous-time counterpart of x_k.

$u_k \in \mathcal{U}(x_k) \subset \mathcal{U}$ is the control or decision variable to be selected at index k in the discrete-time domain. In particular, $u_k = u(t_k)$, i.e., $u(t) = u_k$ for $t_k \leq t < t_{k+1}$, where $u(t)$ describes the continuous-time counterpart of u_k.

$p_{ij}(u)$ is the transition probability from state $x_k = i \in \mathcal{X}$ to state $x_{k+1} = j \in \mathcal{X}$ when the control or decision variable $u_k = u \in \mathcal{U}(i) \subset \mathcal{U}$ is applied, i.e.,

$$p_{ij}(u) = P\{x_{k+1} = j | x_k = i, u_k = u\}. \tag{2.12}$$

$q_{ij}(\tau, u)$ is the Probability Mass Function (PMF) of the transition time τ from state $x_k = i \in \mathcal{X}$ to state $x_{k+1} = j \in \mathcal{X}$ when the control or decision variable $u_k = u \in \mathcal{U}(i) \subset \mathcal{U}$ is applied, i.e.,

$$q_{ij}(\tau, u) = P\{t_{k+1} - t_k = \tau, x_{k+1} = j | x_k = i, u_k = u\}. \tag{2.13}$$

In particular, $q_{ij}(\tau, u)$ can be viewed as a scaled PMF, in the sense that the transition probability $p_{ij}(u)$ multiplies the PMF of the transition time τ, assuming that $p_{ij} > 0$, i.e.,

$$\frac{q_{ij}(\tau, u)}{p_{ij}(u)} = P\{t_{k+1} - t_k = \tau | x_{k+1} = j, x_k = i, u_k = u\}, \tag{2.14}$$

$$p_{ij}(u) = \sum_{\tau=0}^{\infty} q_{ij}(\tau, u). \tag{2.15}$$

Since the transition time τ is a discrete random variable, then the scaled PMF $q_{ij}(\tau, u)$ is a discontinuous and impulse-shaped function.

$Q_{ij}(\tau, u)$ is the Cumulative Distribution Function (CDF) of the transition time τ from state $x_k = i \in \mathcal{X}$ to state $x_{k+1} = j \in \mathcal{X}$ when the control or decision variable $u_k = u \in \mathcal{U}(i) \subset \mathcal{U}$ is applied, i.e.,

$$Q_{ij}(\tau, u) = P\{t_{k+1} - t_k \leq \tau, x_{k+1} = j | x_k = i, u_k = u\}. \tag{2.16}$$

In particular, $Qij(\tau, u)$ can be viewed as a scaled CDF, in the sense that the transition probability $p_{ij}(u)$ multiplies the CDF of the transition time τ, assuming that $p_{ij} > 0$, i.e.,

$$\frac{Q_{ij}(\tau, u)}{p_{ij}(u)} = P\{t_{k+1} - t_k \leq \tau | x_{k+1} = j, x_k = i, u_k = u\}, \tag{2.17}$$

$$p_{ij}(u) = \lim_{\tau \to \infty} Q_{ij}(\tau, u). \tag{2.18}$$

Since the transition time τ is a discrete random variable, then the scaled CDF $Q_{ij}(\tau, u)$ is a discontinuous and staircase-shaped function.

Note that, in our model formulation, the scaled PDF $q_{ij}(\tau, u)$ and the scaled CDF $Q_{ij}(\tau, u)$ replace the transition probability $p_{ij}(u)$.

$\bar{\tau}_i(u)$ is the expected transition time per stage at state $x_k = i \in \mathcal{X}$ when the control or decision variable $u_k = u \in \mathcal{U}(i) \subset \mathcal{U}$ is applied, i.e.,

$$\bar{\tau}_i(u) = \sum_{j \in \mathcal{X}} p_{ij}(u) E\{\tau | i, u, j\}, \tag{2.19}$$

where $E\{\tau | i, u, j\}$ is the conditional expected value of the transition time τ, i.e.,

$$E\{\tau | i, u, j\} = \sum_{\tau=0}^{\infty} \tau \frac{q_{ij}(\tau, u)}{p_{ij}(u)}. \tag{2.20}$$

$G(i,u)$ is the expected cost per stage at state $x_k = i \in \mathcal{X}$ when the control or decision variable $u_k = u \in \mathcal{U}(i) \subset \mathcal{U}$ is applied, i.e.,

$$G(i,u) = \sum_{j \in \mathcal{X}} p_{ij}(u) G(i,u,j), \tag{2.21}$$

where $G(i,u,j)$ is the cost per stage at state $x_k = i \in \mathcal{X}$ when the control or decision variable $u_k = u \in \mathcal{U}(i) \subset \mathcal{U}$ is applied.

$g(i,u)$ is the expected cost per time unit at state $x_k = i \in \mathcal{X}$ when the control or decision variable $u_k = u \in \mathcal{U}(i) \subset \mathcal{U}$ is applied, i.e.,

$$g(i,u) = \sum_{j \in \mathcal{X}} p_{ij}(u) g(i,u,j), \tag{2.22}$$

where $g(i,u,j)$ is the cost per time unit at state $x_k = i \in \mathcal{X}$ when the control or decision variable $u_k = u \in \mathcal{U}(i) \subset \mathcal{U}$ is applied.

In particular,

$$G(i,u,j) = g(i,u,j)\bar{\tau}_i(u). \tag{2.23}$$

2.2 The Markov Decision Chain

The problem of designing the optimal strategies for the scheduling of a Single-Server Multiple-Buffer queueing system is based on the setting of a Markov Decision Chain, where $\mathcal{X}$ is the set of states and $\mathcal{U}$ is the set of control or decision variables.

Let $x_k \in \mathcal{X}$ be the state at index k. In particular, $x_k = (d(k), a(k), b(k), c(k))$, where the variable $d(k) \in \mathcal{D} = \{0, \ldots, D\}$ tracks the number of stages during which $update(k)$ is buffered in HPQ, the variable $update(k) \in \{0, 1\}$ tracks the number of new code-vectors buffered in HPQ, the variable $a(k) \in \mathcal{A} = \{0, \ldots, A\}$ tracks the number of current codebook indexes buffered in LPQ_a, the variable $b(k) \in \mathcal{B} = \{0, \ldots, B\}$ tracks the number of OOS codebook indexes buffered in LPQ_b, and the variable $c(k) \in \mathcal{C} = \{0, \ldots, C\}$ captures the evolution of the multipath Rayleigh fading channel throught a $(C+1)$-state Markov Chain. Note that if $update(k) = 0$, then $d(k) = 0$, whereas if

$update(k) = 1$, then $d(k) > 0$. Furthermore, note that $update(k) \in \{0, 1\}$, whereas $a(k) \in \mathcal{A} = \{0, \ldots, A\}$ with $A > 1$ and $b(k) \in \mathcal{B} = \{0, \ldots, B\}$ with $B > 1$. In other words, we assume that the new code-vectors arrival rate is smaller than the codebook indexes arrival rate, thus the size of HPQ is smaller than the size of LPQ.

Let $u_k \in \mathcal{U}(x_k) \subset \mathcal{U}$ be the control or decision variable to be selected at index k. In particular, $\mathcal{U} = \{HP, LP, IDLE\}$, where the control or decision variable $u_k = HP \in \mathcal{U}(x_k) \subset \mathcal{U}$ stands for a new code-vector transmission from HPQ, the control or decision variable $u_k = LP \in \mathcal{U}(x_k) \subset \mathcal{U}$ stands for a codebook index transmission from LPQ, and the control or decision variable $u_k = IDLE \in \mathcal{U}(x_k) \subset \mathcal{U}$ means that no packet is transmitted at index k. Furthermore, $u_k = HP$ can be chosen $\forall x_k \in \mathcal{X}$ and $\forall k$, such that $update(k) = 1$ (i.e., $u_k = HP$ can be chosen if there is at least one new code-vector in HPQ), $u_k = LP$ can be chosen $\forall x_k \in \mathcal{X}$ and $\forall k$, such that $a(k) \neq 0$ or $a(k) = 0$ but $b(k) \neq 0$ (i.e., $u_k = LP$ can be chosen if there is at least one codebook index in LPQ), and $u_k = IDLE$ can be chosen $\forall x_k \in \mathcal{X}$ and $\forall k$ (i.e., $u_k = IDLE$ is a possible choice $\forall k$). Nevertheless, if $a(k) \neq 0$, then $u_k = LP$ stands for a current codebook index transmission from LPQ_a, whereas if $a(k) = 0$ but $b(k) \neq 0$, then $u_k = LP$ stands for a OOS codebook index transmission from LPQ_b. In other words, we assume that the transmission of the *good codebook indexes* is preferred over the transmission of the *bad codebook indexes*.

According to the codebook-based approaches, we assume that n time units are required for the transmission of the new code-vectors, otherwise just 1 time unit is required for the transmission of the codebook indexes. In other words, the parameter n recalls that the transmission of the new code-vectors is more resource demanding than the transmission of the codebook indexes. Furthermore, the parameter n quantifies the ratio between the lengths of the two data packets.

We define the PMF $q_{ij}(\tau, u)$ of the transition time τ as

$$q_{ij}(\tau, u) = \delta(\tau - n) \cdot \mathbb{1}(u = HP) \tag{2.24}$$

$$q_{ij}(\tau, u) = \delta(\tau - 1) \cdot \mathbb{1}(u \neq HP) \tag{2.25}$$

We define the CDF $Q_{ij}(\tau, u)$ of the transition time τ as

$$Q_{ij}(\tau, u) = \mathbb{H}(\tau - n) \cdot \mathbb{1}(u = HP) \tag{2.26}$$

$$Q_{ij}(\tau, u) = \mathbb{H}(\tau - 1) \cdot \mathbb{1}(u \neq HP) \tag{2.27}$$

In particular, $\delta(\cdot)$ is the Dirac delta function and $\mathbb{H}(\cdot)$ is the Heaviside step function (or the unit step function), such that $\mathbb{H}(y) = \int_{-\infty}^{x} \delta(x)dx$. Furthermore, $\mathbb{1}(\cdot)$ is the indicator function, i.e.,

$$\mathbb{1}(u = HP) = 1 \text{ if } u = HP \tag{2.28}$$

$$\mathbb{1}(u \neq HP) = 1 \text{ if } u \neq HP \tag{2.29}$$

Note that the PMF $q_{ij}(\tau, u)$ of the transition time τ and the CDF $Q_{ij}(\tau, u)$ of the transition time τ do not depend on the state $x_{k+1} = j \in \mathcal{X}$, thus we do not take into account the transition probability $p_{ij}(u)$ as a scaling factor.

Let $\bar{\tau}_i(u)$ be the expected transition time per stage at state $x_k = i \in \mathcal{X}$ when the control or decision variable $u_k = u \in \mathcal{U}(i) \subset \mathcal{U}$ is applied. In particular, once the PMF $q_{ij}(\tau, u)$ of the transition time τ is known, the expected transition time $\bar{\tau}_i(u)$ is computed as

$$\bar{\tau}_i(u) = n \cdot \mathbb{1}(u = HP) \tag{2.30}$$

$$\bar{\tau}_i(u) = \mathbb{1}(u \neq HP) \tag{2.31}$$

Let $G(i, u, j)$ be the cost per stage at state $x_k = i \in \mathcal{X}$ when the control or decision variable $u_k = u \in \mathcal{U}(i) \subset \mathcal{U}$ is applied. In particular, once the expected transition time $\bar{\tau}_i(u)$ is known, the cost per stage $G(i, u, j)$ is computed as

$$G(i, u, j) = g(i, u, j)n \cdot \mathbb{1}(u = HP) \tag{2.32}$$

$$G(i, u, j) = g(i, u, j) \cdot \mathbb{1}(u \neq HP) \tag{2.33}$$

where $g(i, u, j)$ is the cost per time unit at state $x_k = i \in \mathcal{X}$ when the control or decision variable $u_k = u \in \mathcal{U}(i) \subset \mathcal{U}$ is applied.

The problem of designing the optimal strategies follows from the problem of setting the cost model, the channel model, and the transition model, which are introduced next.

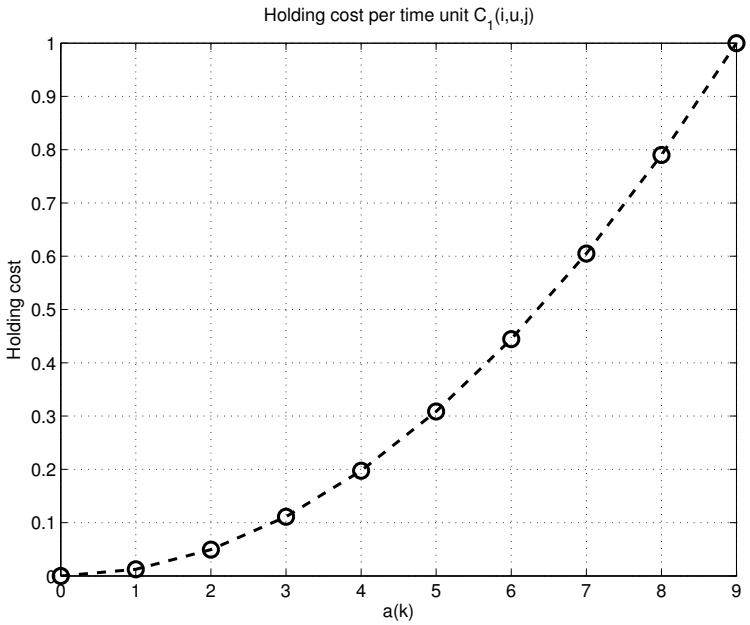

Figure 2.2: Holding cost per time unit $C_1(i, u, j)$ for $A = 9$.

2.2.1 The cost model

According to the theoretical framework considered in [36], the cost per time unit $g(i, u, j)$ is defined as the linear sum of three nonnegative terms, i.e.,

$$g(i, u, j) = \alpha_1 C_1(i, u, j) + \alpha_2 C_2(i, u, j) + \alpha_3 C_3(i, u, j), \tag{2.34}$$

where $C_1(i, u, j)$ is the holding cost per time unit, $C_2(i, u, j)$ is the transmission cost per time unit, $C_3(i, u, j)$ is the OOS transmission cost per time unit, and α_1, α_2, and α_3 are weighting coefficients, such that $\alpha_1 + \alpha_2 + \alpha_3 = 1$. In particular, we assume that $\alpha_1 = 0.25$, $\alpha_2 = 0.25$, $\alpha_3 = 0.5$.

The holding cost per time unit $C_1(i, u, j)$ depends on the number of current codebook indexes buffered in LPQ_a whenever there is no need for the coodebook synchronization, i.e., whenever $update(k) = 0$. In particular, we assume that

$$C_1(i, u, j) = (1/A^2)a(k)^2 \quad a(k) \in \mathcal{A} = \{0, \ldots, A\}. \tag{2.35}$$

In Figure 2.2, we represent the holding cost per time unit $C_1(i, u, j)$ for $A = 9$.

Note that the quadratic function that characterizes the holding cost per time unit $C_1(i, u, j)$ of equation (2.35) tends to introduce a large penalty for large

deviations and a small penalty for small deviations from state $x_k = i \in \mathcal{X}$ to state $x_{k+1} = j \in \mathcal{X}$ as the system evolves along the stochastic trajectory, where the penalty can be regarded as the price paid to postpone the transmission of the *good codebook indexes*.

The transmission cost per time unit $C_2(i, u, j)$ is an additive term that depends on the evolution of the multipath Rayleigh fading channel throught a $(C+1)$-state Markov Chain. In particular, we assume that $C_2(i, u, j) = 0$ if either the trasmission of the codebook index or the trasmission of the new code-vector is successful, whereas $C_2(i, u, j) = 1$ if either the trasmission of the codebook index or the trasmission of the new code-vector is unsuccessful.

The OOS transmission cost per time unit $C_3(i, u, j)$ is an additive term that is taken into account to avoid the transmission of the *bad codebook indexes* whenever $update(k) = 1$ and $u_k = LP$, but $a(k) = 0$. In particular, we assume that $C_3(i, u, j) = 1$.

2.2.2 The channel model

In this subsection, we characterize the received Signal to Noise Ratio (SNR) space according to the received SNR thresholds $0 = \Gamma_0 < \Gamma_1 < \cdots < \Gamma_C < \Gamma_{(C+1)} = \infty$, where Γ is the received SNR random variable. The proposed SNR partition method turns out to be a good estimation of the evolution of the multipath Rayleigh fading channel throught a $(C+1)$-state Markov Chain.

The stationary state probability $\theta_{c(k)}$ when the multipath Rayleigh fading channel is in state $c(k) \in \mathcal{C} = \{0, \ldots, C\}$ for 1 time unit, i.e., when the received SNR Γ is in the interval $[\Gamma_{c(k)}, \Gamma_{c(k)+1})$ $\forall c(k) \in \mathcal{C} = \{0, \ldots, C\}$ for 1 time unit, is computed as

$$\theta_{c(k)} = \int_{\Gamma_{c(k)}}^{\Gamma_{c(k)+1}} p_\Gamma(\gamma) d\gamma = \int_{\Gamma_{c(k)}}^{\Gamma_{c(k)+1}} (1/\gamma_0) e^{-\gamma/\gamma_0} d\gamma, \tag{2.36}$$

where $p_\Gamma(\gamma)$ is the exponential PDF of the received SNR Γ and γ_0 is the expected value of the received SNR Γ. In particular, $\gamma_0 = P_{rx}/(N_0 B)$, where P_{rx} is the average power at the receiving side and $N_o B$ is the noise power at the receiving side. Furthermore, according to the Simplified Path Loss Model,

$P_{rx} = P_{tx}K(d_0/d)^{\nu}$, where K is a constant that depends on the antenna at the transmitting side and on the antenna at the receiving side, d is the distance between the transmitter and the receiver, ν is the Path Loss exponent, and P_{tx} is the transmission power of the transmitting biomedical device.

The proposed SNR partition method aims at characterizing the received SNR Γ in the worst performing state $c(k) = 0$, i.e., when the received SNR Γ is in the interval $[\Gamma_0, \Gamma_1)$, and in the best performing state $c(k) = C$, i.e., when the received SNR Γ is in the interval $[\Gamma_C, \Gamma_{(C+1)})$. For this reason, we first introduce two further thresholds, i.e., $p_{thr,1}$ close to 1 and $p_{thr,C}$ close to 0, such that $\Gamma_1 = \mathcal{F}_P^{-1}(p_{thr,1})$ and $\Gamma_C = \mathcal{F}_P^{-1}(p_{thr,C})$. Then, we compute θ_0 and θ_C, given that Γ_0 and $\Gamma_{(C+1)}$ are known. At this point, the criterion to define the channel model when the received SNR Γ is in the interval $[\Gamma_{c(k)}, \Gamma_{c(k)+1})$ $\forall c(k) \in \{1, \ldots, C-1\}$ consists in setting $\theta_{c(k)} = (1 - \theta_0 - \theta_C)/(C-1)$ $\forall c(k) \in \{1, \ldots, C-1\}$ and evaluating the received SNR thresholds from equation (2.36) in a recursive fashion, given that Γ_1 and Γ_C are known.

The codebook index error probability $P_{KO,c(k)}^{(1)}$ when the multipath Rayleigh fading channel is in state $c(k) \in \mathcal{C} = \{0, \ldots, C\}$ for 1 time unit, i.e., when the received SNR Γ is in the interval $[\Gamma_{c(k)}, \Gamma_{c(k)+1})$ $\forall c(k) \in \mathcal{C} = \{0, \ldots, C\}$ for 1 time unit, is computed as

$$P_{KO,c(k)}^{(1)} = \frac{\int_{\Gamma_{c(k)}}^{\Gamma_{c(k)+1}} \mathcal{F}_P(\gamma) p_\Gamma(\gamma) d\gamma}{\theta_{c(k)}} = \frac{\int_{\Gamma_{c(k)}}^{\Gamma_{c(k)+1}} \mathcal{F}_P(\gamma)(1/\gamma_0) e^{-\gamma/\gamma_0} d\gamma}{\theta_{c(k)}}, \tag{2.37}$$

where $\mathcal{F}_P(\gamma) = 1 - (1 - \mathcal{F}_B(\gamma))^L$ is the Packet Error Rate (PER), $\mathcal{F}_B(\gamma)$ is the Bit Error Rate (BER), and L is the codebook index length. In particular, we assume that the channel model deals with a $\pi/4$-DQPSK modulation scheme, thus $\mathcal{F}_B(\gamma)$ can be approximated as $(4/3)\text{erfc}(\sqrt{\gamma})$ [38] [39] [40].

The codebook index success probability $P_{OK,c(k)}^{(1)}$ when the multipath Rayleigh fading channel is in state $c(k) \in \mathcal{C} = \{0, \ldots, C\}$ for 1 time unit, i.e., when the received SNR Γ is in the interval $[\Gamma_{c(k)}, \Gamma_{c(k)+1})$ $\forall c(k) \in \mathcal{C} = \{0, \ldots, C\}$ for 1 time unit, is computed as

$$P_{OK,c(k)}^{(1)} = 1 - P_{KO,c(k)}^{(1)}. \tag{2.38}$$

The channel transition probability $t_{c(k)c(k+1)}^{(1)}$ from state $c(k) \in \mathcal{C} = \{0, \ldots, C\}$

to state $c(k+1) \in \mathcal{C} = \{0, \ldots, C\}$ in 1 time unit is computed as

$$t^{(1)}_{c(k)c(k+1)} = \frac{\int_{\zeta_{c(k+1)}}^{\zeta_{c(k+1)+1}} \int_{\zeta_{c(k)}}^{\zeta_{c(k)+1}} f_{R_1R_2}(r_1, r_2, r) dr_1 dr_2}{\theta_{c(k)}}, \tag{2.39}$$

where $\zeta_{c(k)} = \sqrt{\Gamma_{c(k)}/\gamma_0}$ and $\rho = J_0(2\pi f_d T_p)$. Furthermore, f_d is the Doppler frequency, T_p is the codebook index transmission duration, $J_0(\cdot)$ is the Bessel function of the first kind and order zero, $I_0(\cdot)$ is the modified Bessel function of the first kind and order zero, and $f_{R_1R_2}(r_1, r_2)$ is the bivariate Rayleigh joint pdf, i.e.,

$$f_{R_1R_2}(r_1, r_2) = \frac{4r_1r_2}{1-\rho^2} e^{-(r_1^2+r_2^2)/(1-\rho^2)} I_0(2r_1r_2\rho/(1-\rho^2)). \tag{2.40}$$

Let $t^{(n)}_{c(k)c(k+1)}$ denote the channel transition probability from state $x_k = i \in \mathcal{X}$ to state $x_{k+1} = j \in \mathcal{X}$ when the control or decision variable $u_k = u \in \mathcal{U}(i) \subset \mathcal{U}$ is applied for the expected transition time $\bar{\tau}_i(u) = n$ time units. In particular, the channel transition probability $t^{(n)}_{c(k)c(k+1)}$ $\forall c(k) \in \mathcal{C} = \{0, \ldots, C\}$ and $\forall c(k+1) \in \mathcal{C} = \{0, \ldots, C\}$ takes into account all possibile paths in the form $[t^{(1)}_{c(k)l(1)} t^{(1)}_{l(1)l(2)} \cdots t^{(1)}_{l(n-1)c(k+1)}]$, where $l(m) \in \mathcal{C} = \{0, \ldots, C\}$ $\forall m \in \{1, \ldots, n-1\}$. For the sake of simplicity, we denote $t^{(1)}_{c(k)c(k+1)} = t^{(1)}_{ij}$ and $t^{(n)}_{c(k)c(k+1)} = t^{(n)}_{ij}$, where $i = (d(k), a(k), b(k), c(k))$ and $j = (d(k+1), a(k+1), b(k+1), c(k+1))$.

Let $P^{(n)}_{OK,c(k)} = 1 - P^{(n)}_{KO,c(k)}$ denote the new code-vector success probability of the channel model from state $x_k = i \in \mathcal{X}$ to state $x_{k+1} = j \in \mathcal{X}$ when the control or decision variable $u_k = u \in \mathcal{U}(i) \subset \mathcal{U}$ is applied for the expected transition time $\bar{\tau}_i(u) = n$ time units. In particular, the new code-vector success probability $P^{(n)}_{OK,c(k)}$ takes into account all possible paths in the form $[P^{(1)}_{OK,c(k)} P^{(1)}_{OK,l(1)} \cdots P^{(1)}_{OK,l(n-1)}] \cdot [t^{(1)}_{c(k)l(1)} t^{(1)}_{l(1)l(2)} \cdots t^{(1)}_{l(n-2)l(n-1)}]$, where $l(m) \in \mathcal{C} = \{0, \ldots, C\}$ $\forall m \in \{1, \ldots, n-1\}$. For the sake of simplicity, we denote $P^{(1)}_{OK,c(k)} = P^{(1)}_{OK,ij}$ and $P^{(n)}_{OK,c(k)} = P^{(n)}_{OK,ij}$, where $i = (d(k), a(k), b(k), c(k))$ and $j = (d(k+1), a(k+1), b(k+1), c(k+1))$, although $P^{(1)}_{OK,c(k)}$ and $P^{(n)}_{OK,c(k)}$ do not depend on $c(k+1)$.

Note that $(C+1)^{(n-1)}$ is the number of possible paths from state $x_k = i \in \mathcal{X}$ to state $x_{k+1} = j \in \mathcal{X}$ for the expected transition time $\bar{\tau}_i(u) = n$ time units. As the value of n increases, then the number of all possible paths increases as well, thus leading to a smaller $P^{(n)}_{OK,c(k)}$.

2.2.3 The transition model

We assume that the new code-vectors arrival process and the codebook indexes arrival process are Bernoulli distributed. Let λ_H and λ_L respectively denote the new code-vectors arrival rate in HPQ and the codebook indexes arrival rate in LPQ. Let $\bar{\tau}$ denote the length of a time unit. It follows that

$$p_H = P\{1 \text{ new code-vector arrival in 1 time unit}\} = \lambda_H \bar{\tau} \tag{2.41}$$

$$p_L = P\{1 \text{ codebook index arrival in 1 time unit}\} = \lambda_L \bar{\tau} \tag{2.42}$$

In general, $\forall m \in \{0, \ldots, n\}$

$$\begin{aligned} p_H(m,n) &= P\{m \text{ new code-vectors arrivals in } n \text{ time units}\} \\ &= \binom{n}{m}(1-p_H)^{(n-m)}p_H^m \end{aligned} \tag{2.43}$$

$$\begin{aligned} p_L(m,n) &= P\{m \text{ codebook indexes arrivals in } n \text{ time units}\} \\ &= \binom{n}{m}(1-p_L)^{(n-m)}p_L^m \end{aligned} \tag{2.44}$$

thus

$$p_H(1,1) = p_H \quad \text{and} \quad p_H(0,1) = (1-p_H) \tag{2.45}$$

$$p_L(1,1) = p_L \quad \text{and} \quad p_L(0,1) = (1-p_L) \tag{2.46}$$

To be clear about the transition model, some assumptions are introduced next.

In general, we assume that the transition model accounts for 0 packet arrivals in 1 time unit w.p. $p_H(0,1)p_L(0,1)$, 1 packet arrival in 1 time unit w.p. $p_H(1,1)p_L(0,1)$ or w.p. $p_H(0,1)p_L(1,1)$, or 2 packet arrivals in 1 time unit w.p. $p_H(1,1)p_L(1,1)$. In the case of 2 packet arrivals in 1 time unit w.p. $p_H(1,1)p_L(1,1)$, we assume that the new code-vector arrival preceeds the codebook index arrival, thus the codebook index that has arrived w.p. $p_L(1,1)$ is considered OOS and buffered in LPQ_b.

$a(k+1) = [x]^+$ and $b(k+1) = [x]^+$ stand for $a(k+1) = \max(x, 0)$ and $b(k+1) = \max(x, 0)$, whereas $a(k+1) = [x]^-$ and $b(k+1) = [x]^-$ stand for $a(k+1) = \min(x, A)$ and $b(k+1) = \min(x, B)$. Also, $d(k+1) = [x]^-$ stands for $d(k+1) = \min(x, D)$.

If $update(k) = 0$, then $b(k) = 0$. In other words, we assume that, if there is no new code-vector waiting for transmission, then there are no OOS codebook indexes waiting for transmission as well. Let $\mathcal{X}_0$ and $\mathcal{X}_1$ be the defined as the subsets of $\mathcal{X}$ such that $x_k \in \mathcal{X}_0 \subset \mathcal{X}$ whenever $update(k) = 0$, whereas $x_k \in \mathcal{X}_1 \subset \mathcal{X}$ whenever $update(k) = 1$. It follows that $x_k = (d(k) = 0, a(k), b(k) = 0, c(k)) \in \mathcal{X}_0 \subset \mathcal{X}$ and $x_k = (d(k) > 0, a(k), b(k), c(k)) \in \mathcal{X}_1 \subset \mathcal{X}$, thus $|\mathcal{X}_0| = (A+1)(C+1)$ and $|\mathcal{X}_1| = D(A+1)(B+1)(C+1)$. Furthermore, $|\mathcal{X}| = |\mathcal{X}_0| + |\mathcal{X}_1|$, i.e., the cardinality of $\mathcal{X}$ is equal to the sum of the cardinalities of $\mathcal{X}_0$ and $\mathcal{X}_1$.

Given $update(k) = 0$ and $x_k = (d(k) = 0, a(k), b(k) = 0, c(k))$, the transition model is defined as follows.

If $u_k = LP$, then $x_{k+1} =$

$(0, [a(k) - 1]^+, 0, c(k+1))$	w.p. $p_H(0,1)p_L(0,1)t_{ij}^{(1)}P_{OK,ij}^{(1)}$
$(0, a(k), 0, c(k+1))$	w.p. $p_H(0,1)p_L(0,1)t_{ij}^{(1)}P_{KO,ij}^{(1)}$
$(1, [a(k) - 1]^+, 0, c(k+1))$	w.p. $p_H(1,1)p_L(0,1)t_{ij}^{(1)}P_{OK,ij}^{(1)}$
$(1, a(k), 0, c(k+1))$	w.p. $p_H(1,1)p_L(0,1)t_{ij}^{(1)}P_{KO,ij}^{(1)}$
$(0, [a(k) - 1]^+ + 1, 0, c(k+1))$	w.p. $p_H(0,1)p_L(1,1)t_{ij}^{(1)}P_{OK,ij}^{(1)}$
$(0, [a(k) + 1]^-, 0, c(k+1))$	w.p. $p_H(0,1)p_L(1,1)t_{ij}^{(1)}P_{KO,ij}^{(1)}$
$(1, [a(k) - 1]^+, 1, c(k+1))$	w.p. $p_H(1,1)p_L(1,1)t_{ij}^{(1)}P_{OK,ij}^{(1)}$
$(1, a(k), 1, c(k+1))$	w.p. $p_H(1,1)p_L(1,1)t_{ij}^{(1)}P_{KO,ij}^{(1)}$

If $u_k = IDLE$, then $x_{k+1} =$

$(0, a(k), 0, c(k+1))$	w.p. $p_H(0,1)p_L(0,1)t_{ij}^{(1)}$
$(1, a(k), 0, c(k+1))$	w.p. $p_H(1,1)p_L(0,1)t_{ij}^{(1)}$
$(0, [a(k) + 1]^-, 0, c(k+1))$	w.p. $p_H(0,1)p_L(1,1)t_{ij}^{(1)}$
$(1, a(k), 1, c(k+1))$	w.p. $p_H(1,1)p_L(1,1)t_{ij}^{(1)}$

If $update(k) = 1$, then the transmitter and the receiver are said to be OOS, waiting for the new code-vector synchronization. If $update(k) = 1$ and $u_k \neq HP$ is chosen, then the transmitter and the receiver are again OOS after just 1 time unit and $d(k+1) = [d(k) + 1]^-$ w.p. 1, thus the codebook index that has

arrived w.p. $p_L(1,1)$ is cosidered OOS and buffered in LPQ$_b$. If $update(k) = 1$ and $u_k = HP$, then the transmitter and the receiver are again OOS after n time units and $d(k+1) = [d(k)+1]^-$ w.p. $P^{(n)}_{KO,ij}$, thus the $m \in \{1, \ldots, n-1\}$ codebook indexes that have arrived in these n time units w.p. $p_L(m,n)$ are cosidered OOS and buffered in LPQ$_b$. If $update(k) = 1$ and $u_k = HP$, the transmitter and the receiver are not OOS after n time units and $d(k+1) = 0$ w.p. $P^{(n)}_{OK,ij}$, thus the $m \in \{1, \ldots, n-1\}$ codebook indexes that have arrived in these n time units w.p. $p_L(m,n)$ are considered not OOS and buffered in LPQ$_a$, along with both $a(k)$ and $b(k)$, such that $a(k+1) = [a(k)+b(k)+m]^-$.

If $update(k) = 1$, then the new code-vectors arrival process is turned off until the new code-vector transmission in completed. If $update(k) = 1$ and $u_k \neq HP$ is chosen, then $update(k+1) = 1$ and $d(k+1) = [d(k)+1]^-$ w.p. 1. If $update(k) = 1$ and $u_k = HP$, then $update(k+1) = 1$ and $d(k+1) = [d(k)+1]^-$ w.p. $P^{(n)}_{KO,ij}$ or $update(k+1) = 0$ and $d(k+1) = 0$ w.p. $P^{(n)}_{OK,ij}$. This is not the case for the codebook indexes arrival process.

Given $update(k) = 1$ and $x_k = (0 < d(k) < D, a(k), b(k), c(k))$, the transition model is defined as follows.

If $u_k = HP$, then $x_{k+1} =$

$(0, [a(k)+b(k)+m]^-, 0, c(k+1))$ w.p. $p_L(m,n)t^{(n)}_{ij}P^{(n)}_{OK,ij}$

$([d(k)+1]^-, a(k), [b(k)+m]^-, c(k+1))$ w.p. $p_L(m,n)t^{(n)}_{ij}P^{(n)}_{KO,ij}$

If $u_k = LP$ and $a = 0$, then $x_{k+1} =$

$([d(k)+1]^-, 0, [b(k)-1]^+, c(k+1))$ w.p. $p_L(0,1)t^{(1)}_{ij}P^{(1)}_{OK,ij}$

$([d(k)+1]^-, 0, b(k), c(k+1))$ w.p. $p_L(0,1)t^{(1)}_{ij}P^{(1)}_{KO,ij}$

$([d(k)+1]^-, 0, [b(k)-1]^+ + 1, c(k+1))$ w.p. $p_L(1,1)t^{(1)}_{ij}P^{(1)}_{OK,ij}$

$([d(k)+1]^-, 0, [b(k)+1]^-, c(k+1))$ w.p. $p_L(1,1)t^{(1)}_{ij}P^{(1)}_{KO,ij}$

If $u_k = LP$ and $a \neq 0$, then $x_{k+1} =$

$([d(k)+1]^-, [a(k)]^+, b(k), c(k+1))$ w.p. $p_L(0,1)t^{(1)}_{ij}P^{(1)}_{OK,ij}$

$([d(k)+1]^-, a(k), b(k), c(k+1))$ w.p. $p_L(0,1)t^{(1)}_{ij}P^{(1)}_{KO,ij}$

$([d(k)+1]^-, [a(k)-1]^+, [b(k)+1]^-, c(k+1))$ w.p. $p_L(1,1)t^{(1)}_{ij}P^{(1)}_{OK,ij}$

$([d(k)+1]^-, a(k), [b(k)+1]^-, c(k+1))$ w.p. $p_L(1,1)t^{(1)}_{cij}P^{(1)}_{KO,ij}$

If $u_k = IDLE$, then $x_{k+1} =$

$([d(k)+1]^-, a(k), b(k), c(k+1))$ w.p. $p_L(0,1)t_{ij}^{(1)}$

$([d(k)+1]^-, a(k), [b(k)+1]^-, c(k+1))$ w.p. $p_L(1,1)t_{ij}^{(1)}$

In general, if $d(k) = 0$, then we assume that there are 2 available control or decision variables, i.e., $u_k = LP$ or $u_k = IDLE$ (except for the case $x_k = (d(k) = 0, 0, 0, c(k))$, where the control or decision variable $u_k = LP$ cannot be chosen), whereas if $0 < d(k) < D$, then we assume that there are 3 available control or decision variables, i.e., $u_k = HP$, $u_k = LP$ or $u_k = IDLE$ (except for the case $x_k = (0 < d(k) < D, 0, 0, c(k))$, where the control or decision variable $u_k = LP$ cannot be chosen).

If $d(k) = D$, then we assume that there is just 1 possible control or decision variable, i.e., $u_k = HP$. In other words, if $d(k) = D$, then our model formulation forces the new code-vector synchronization without looking at the evolution of the multipath Rayleigh fading channel throught a $(C+1)$-state Markov Chain.

Chapter 3

Performance evaluation

The DP and Optimal Control theory aims at minimizing the average cost of equation (2.10) taking advantage of the Value Iteration Algorithm, thus finding the optimal average cost and the optimal policy of the infinite time horizon, average cost, semi-Markov problem, with three further important considerations. First, this mathematical tools allow us to determine how the optimal average cost and the optimal policy depend on the system parameters and on the evolution of the multipath Rayleigh fading channel. Second, this mathematical tools allow us to evaluate the system performance under the optimal policy, according to which the central controller has full knowledge of the system parameters and of the evolution of the multipath Rayleigh fading channel. Third, we can compare the system performance under the optimal policy with the system performance under two suboptimal policies (also referred to as heuristic policies H_1 and H_2), which do not require full state information and turn out to be computationally light.

3.1 The Value Iteration Algorithm

We compare the original version of the Value Iteration Algorithm (Algorithm 1) with the modified version of the Value Iteration Algorithm (Algorithm 2). Furthermore, the modified version of the Value Iteration Algorithm (Algorithm 2) is presented to suit our model formulation, i.e., to suit the infinite time horizon, average cost, semi-Markov problem.

Set $k = 0$, $\epsilon > 0$, $N > 0$. Initialize $J_0(i)$ $\forall i \in \mathcal{X}$.
while $k < N$ **do**
  for $i \in \mathcal{X}$ **do**
    for $j \in \mathcal{X}$ **do**
      $J_{k+1}(i) = \min_{u \in \mathcal{U}(i) \subset \mathcal{U}} p_{ij}(u)(g(i,u,j) + J_k(j))$
      $\pi_{k+1}(i) = \arg\min_{u \in \mathcal{U}(i)} p_{ij}(u)(g(i,u,j) + J_k(j))$
    end
  end
  if $sp(J_k, J_{k+1}) < \epsilon$ **then**
    Condition verified
  end
  $k = k + 1$
end

Algorithm 1: The Value Iteration Algorithm (original version)

Set $k = 0$, $\epsilon > 0$, $N > 0$. Initialize $J_0(i)$ $\forall i \in \mathcal{X}$. Initialize $t_0(i)$ $\forall i \in \mathcal{X}$.
while $k < N$ **do**
  for $i \in \mathcal{X}$ **do**
    for $j \in \mathcal{X}$ **do**
      $J_{k+1}(i) = \min_{u \in \mathcal{U}(i) \subset \mathcal{U}} \frac{p_{ij}(u)(G(i,u,j) + J_k(j)t_k(j))}{p_{ij}(u)t_k(j) + n \cdot 1(u = HP) + 1(u \neq HP)}$
      $\pi_{k+1}(i) = \arg\min_{u \in \mathcal{U}(i)} \frac{p_{ij}(u)(G(i,u,j) + J_k(j)t_k(j))}{p_{ij}(u)t_k(j) + n \cdot 1(u = HP) + 1(u \neq HP)}$
    end
  end
  $t_{k+1}(i) = p_{ij}(u)t_k(j) + n \cdot 1(u = HP) + 1(u \neq HP)$
  if $sp(J_k, J_{k+1}) < \epsilon$ **then**
    Condition verified
  end
  $k = k + 1$
end

Algorithm 2: The Value Iteration Algorithm (modified version)

The original version of the Value Iteration Algorithm (Algorithm 1) aims at minimizing the expected total cost $J_{k+1}(i)$ $\forall i \in \mathcal{X}$, computed adding $g(i,u,j)$ to $J_k(j)$ $\forall j \in \mathcal{X}$, such that $p_{ij}(u) \neq 0$, where $g(i,u,j)$ is the cost per time unit at state $x_k = i \in \mathcal{X}$ when the control or decision variable $u_k = u \in \mathcal{U}(i) \subset \mathcal{U}$ is applied. In particular, we assume that $J_0(i) = 0$ $\forall i \in \mathcal{X}$.

The modified version of the Value Iteration Algorithm (Algorithm 2) aims at minimizing the expected total cost $J_{k+1}(i)t_{k+1}(i)\ \forall i \in \mathcal{X}$, computed adding $G(i,u,j)$ to $J_k(j)t_k(j)\ \forall j \in \mathcal{X}$, such that $p_{ij}(u) \neq 0$, $G(i,u,j)$ is the cost per stage at state $x_k = i \in \mathcal{X}$ when the control or decision variable $u_k = u \in \mathcal{U}(i) \subset \mathcal{U}$ is applied. In particular, we assume that $J_0(i) = 0\ \forall i \in \mathcal{X}$. Furthermore, $t_k(j)\ \forall j \in \mathcal{X}$ is the expected completion time of the k-th decision epoch.

Let $J^*_{k+1}(i)\ \forall i \in \mathcal{X}$ denote the optimal average cost resulting at the end of the Value Iteration Algorithm. In particular, the optimal average cost $J^*_{k+1}(i)$ $\forall i \in \mathcal{X}$ depends on the precision parameter $\epsilon > 0$. Nevertheless, the difference in performance between $J_{k+1}(i)\ \forall i \in \mathcal{X}$ and $J_k(i)\ \forall i \in \mathcal{X}$ becomes negligible as ϵ tends to zero, according to the span seminorm operator defined as

$$sp(J_k, J_{k+1}) = \max_{i \in \mathcal{X}}(J_{k+1}(i) - J_k(i)) - \min_{i \in \mathcal{X}}(J_{k+1}(i) - J_k(i)) \tag{3.1}$$

Let $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$ denote the optimal policy resulting at the end of the Value Iteration Algorithm. Note that, once the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$ is known, then the Markov Decision Chain (also known as a discrete-time Markov Decision Process) turns out to be a Markov Chain. As a consequence, the stationary state probability $\rho_i\ \forall i \in \mathcal{X}$ of the model transition matrix induced by the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$ is computed as

$$\sum_{j \in \mathcal{X}} \rho_j \cdot m_{ij} = \rho_i \quad \forall i \in \mathcal{X} \quad \text{and} \quad \sum_{i \in \mathcal{X}} \rho_i = 1 \tag{3.2}$$

where m_{ij} is the element on the i-th row and on the j-th column of the model transition matrix induced by the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$.

The heuristic policies H_1 and H_2 are defined as follows.

The heuristic policy H_1 applies the control or decision variable $u_k = HP$ whenever possible (i.e., whenever $update(k) > 0$) without looking at the evolution of the multipath Rayleigh fading channel throught a $(C+1)$-state Markov Chain, thus taking into account some forced transmission cost. Otherwise, it applies the control or decision variable $u_k = IDLE$. According to our model formulation, if $d(k) = D$, then assume that there is just 1 possible control or decision variable, i.e., $u_k = HP$.

The heuristic policy H_2 applies the control or decision variable $u_k = LP$ whenever possible (i.e., whenever $a(k) > 0$ or $b(k) > 0$) without looking at the evolution of the multipath Rayleigh fading channel throught a $(C+1)$-state Markov Chain, thus taking into account some forced transmission cost. Otherwise, it applies the control or decision variable $u_k = IDLE$. According to our model formulation, if $d(k) = D$, then assume that there is just 1 possible control or decision variable, i.e., $u_k = HP$.

We proceed with the computation and the discussion of the following metrics of interest due to the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$: the average long term cost C_π, the percentage of discarded codebook indexes (both potential *good codebook indexes* and potential *bad codebook indexes*), the energy efficiency of $u_k = HP$ or $u_k = LP$, which will be defined as the percentage of successful channel uses due to the control or decision variable $u_k = HP$ or $u_k = LP$, respectively, and the average new code-vector delay. The metrics due to the heuristic policies H_1 and H_2 are obtained using the same reasoning.

3.2 Numerical results

We investigate the numerical results for different values of the transmission power of the transmitting biomedical device and for different values of the new code-vectors arrival rate. In particular, we assume that $P_{tx} \in \{100, 500\}$ mW and $\lambda_H \in [0.005, 0.5]$ new code-vectors arrivals/s. Furthermore, we consider $D = 3$, $A = 9$, $B = 9$, $C = 3$, $n \in \{1, 5, 10\}$ time units, $\bar{\tau} = 1$ s, $\lambda_L = 0.5$ codebook indexes arrivals/s, thus $p_L = 0.5$ codebook indexes. In the following plots, the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$ is referred to as OPT.

In Figure 3.1 and Figure 3.2, we represent the PER in semi-logarithmic scale and the received SNR thresholds $0 = \Gamma_0 < \Gamma_1 < \cdots < \Gamma_C < \Gamma_{(C+1)} = \infty$ (where $\Gamma_{(C+1)} = \infty$ is not plotted) for $C = 3$. In particular, Figure 3.1 reports the received SNR thresholds when $P_{tx} = 100$ mW, whereas Figure 3.2 reports the received SNR thresholds when $P_{tx} = 500$ mW. Furthermore, we assume that $p_{thr,1} = 10^{-1}$ and $p_{thr,C} = 10^{-10}$.

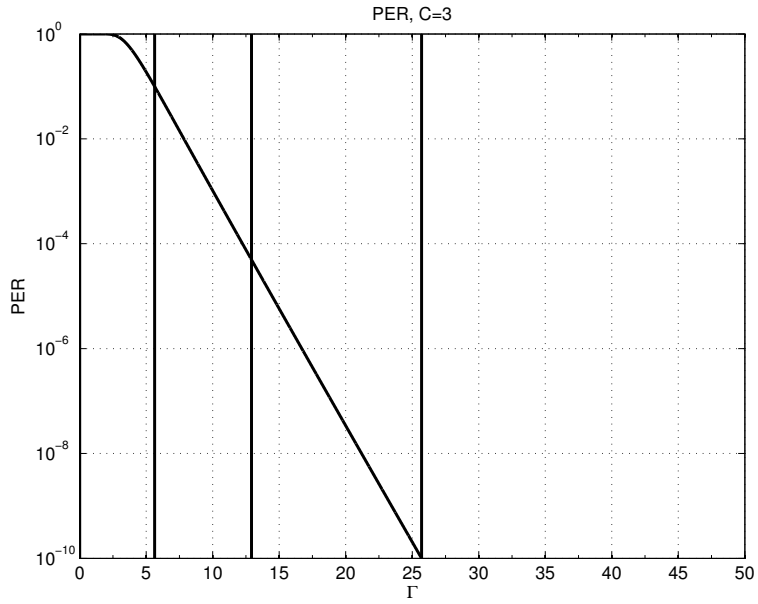

Figure 3.1: PER in semi-logarithmic scale for $P_{tx} = 100$ mW.

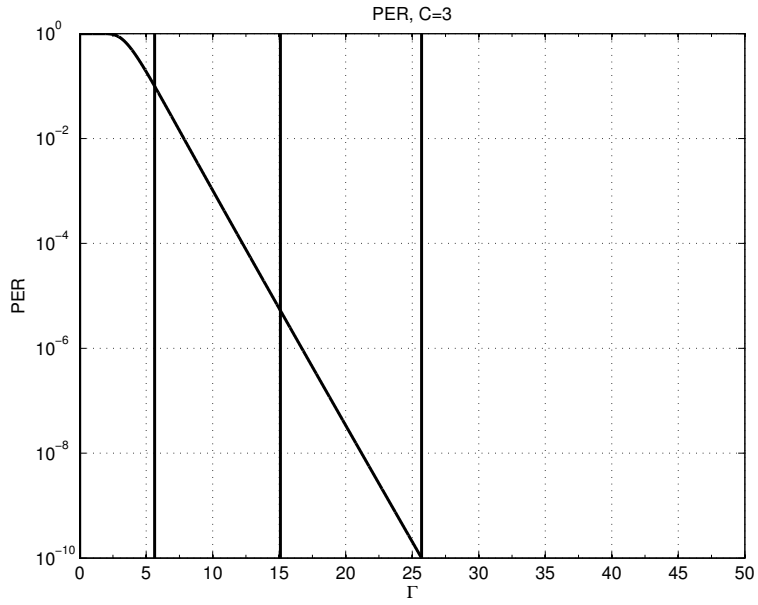

Figure 3.2: PER in semi-logarithmic scale for $P_{tx} = 500$ mW.

According to the different values of the transmission power of the transmitting biomedical device, Figure 3.1 and Figure 3.2 present different computations of the received SNR thresholds, thus leading to different settings of the channel model. Note that, if these settings of the channel model are good enough, i.e., if the higher codebook index error probabilitity $P^{(1)}_{KO,c(k)}$ $\forall c(k) \in \mathcal{C} = \{0, \ldots, C\}$ is associated to the smaller stationary state probability $\theta_{c(k)}$ when the multipath Rayleigh fading channel is in state $c(k) \in \mathcal{C} = \{0, \ldots, C\}$ for 1 time unit, then it might happen that the heuristic policies H_1 and H_2 outperform the optimal policy $\pi^*_{k+1}(i)$ $\forall i \in \mathcal{X}$, or at least approach it, as we will see.

In Figure 3.3 and Figure 3.4, we represent the behavior of the optimal policy $\pi^*_{k+1}(i)$ $\forall i \in \mathcal{X}$ as a function of $a(k) \in \mathcal{A} = \{0, \ldots, A\}$ and $b(k) \in \mathcal{B} = \{0, \ldots, B\}$ when the multipath Rayleigh fading channel is in the best performing state $c(k) = C$. In these plots, the red squared markers stand for the control or decision variable $u_k = HP$, the blue circular markers stand for the control or decision variable $u_k = LP$, and the green upward-pointing triangular markers stand for the control or decision variable $u_k = IDLE$. In particular, Figure 3.3 reports the behavior of the optimal policy $\pi^*_{k+1}(i)$ $\forall i \in \mathcal{X}$ when $n = 1$ time unit, whereas Figure 3.4 reports the behavior of the optimal policy $\pi^*_{k+1}(i)$ $\forall i \in \mathcal{X}$ when $n = 10$ time units. Furthermore, we consider $P_{tx} = 500$ mW and $\lambda_H = 0.005$ new code-vectors arrivals/s.

Since $update(k) = 1$, the transmitter and the receiver are waiting for the new code-vector synchronization. In particular, if the control or decision variable $u_k = HP$ is chosen, then the transmitter and the receiver are not OOS after n time units and $d(k+1) = 0$ w.p. $P^{(n)}_{OK,ij}$, according to the multipath Rayleigh fading channel. In this case, it might happen that $a(k+1) = [a(k)+b(k)+m]^-$, where the $m \in \{1, \ldots, n-1\}$ codebook indexes that have arrived in these n time units w.p. $p_L(m, n)$ are considered not OOS and buffered in LPQ_a, along with both $a(k)$ and $b(k)$. Nevertheless, note that $a(k+1) = [a(k)+b(k)+m]^-$ is bounded between 0 and A.

The behavior of the optimal policy $\pi^*_{k+1}(i)$ $\forall i \in \mathcal{X}$ results in the attempted anticipation of the new code-vector synchronization whenever the multipath Rayleigh fading channel is in the best performing state $c(k) = C$, or at least

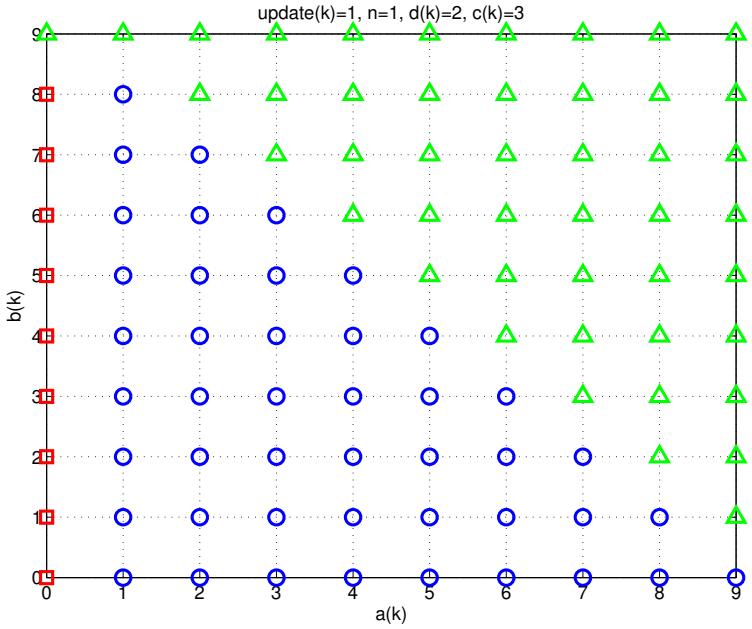

Figure 3.3: Optimal policy representation for $n = 1$.

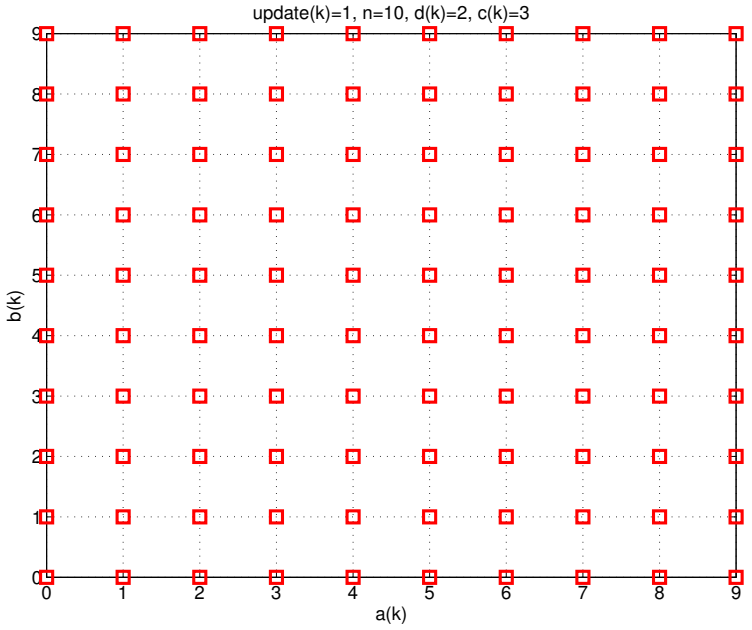

Figure 3.4: Optimal policy representation for $n = 10$.

in a good enough performing state. According to the holding cost per time unit $C_1(i,u,j)$ of equation (2.35), the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$ suggests the central controller to choose $u_k = LP$ as long as the holding cost per time unit $C_1(i,u,j)$ at index $k+1$ is smaller than or equal to the holding cost per time unit $C_1(i,u,j)$ at index k, i.e., as long as $a(k+1)$ is smaller than or equal to $a(k)$. Otherwise, the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$ suggests the central controller to choose $u_k = IDLE$, since $a(k+1) = [a(k) + b(k) + m]^-$ is greater than A, thus the codebook indexes are dropped from LPQ because of the finite capacities of the buffers. Also, the OOS transmission cost per time unit $C_3(i,u,j)$ is taken into account to avoid the OOS codebook indexes transmission. These considerations hold even more as the value of n increases from $n = 1$ time unit to $n = 10$ time units. In Figure 3.4, it is evident that there is no more dependence of the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$ on the value of $a(k) \in \mathcal{A} = \{0, \ldots, A\}$ and $b(k) \in \mathcal{B} = \{0, \ldots, B\}$.

3.2.1 The average long term cost C_π

The average long term cost C_π due to the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$ is computed as

$$C_\pi = \sum_{i \in \mathcal{X}} \rho_i \cdot J^*_{k+1}(i) \tag{3.3}$$

where the stationary state probability $\rho_i\ \forall i \in \mathcal{X}$ of the model transition matrix induced by the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$ follows from equation (3.2).

In Figure 3.5 and Figure 3.6, we represent the average long term cost due to the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$ and due to the heuristic policies H_1 and H_2 as a function of $\lambda_H \in [0.005, 0.5]$ new code-vectors arrivals/s. In these plots, the red squared markers stand for the optiml policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$, the blue circular markers stand for the heuristic policy H_1, and the green upward-pointing triangular markers stand the heuristic policy H_2. In particular, Figure 3.5 reports the average long term cost due to the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$ and due to the heuristic policies H_1 and H_2 when $n = 1$ time unit, whereas Figure 3.6 reports the average long term cost due to the optimal policy $\pi^*_{k+1}(i)$ $\forall i \in \mathcal{X}$ and due to the heuristic policies H_1 and H_2 when $n = 10$ time units.

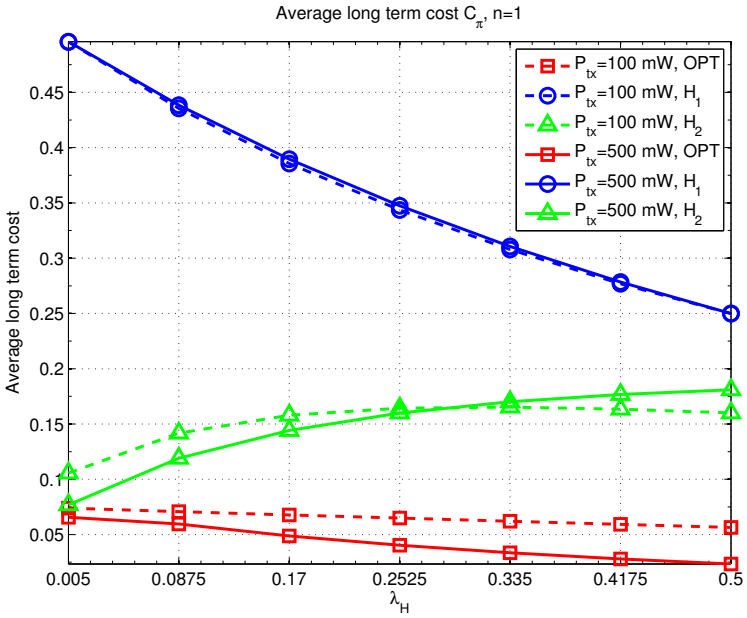

Figure 3.5: Average long term cost for $n = 1$.

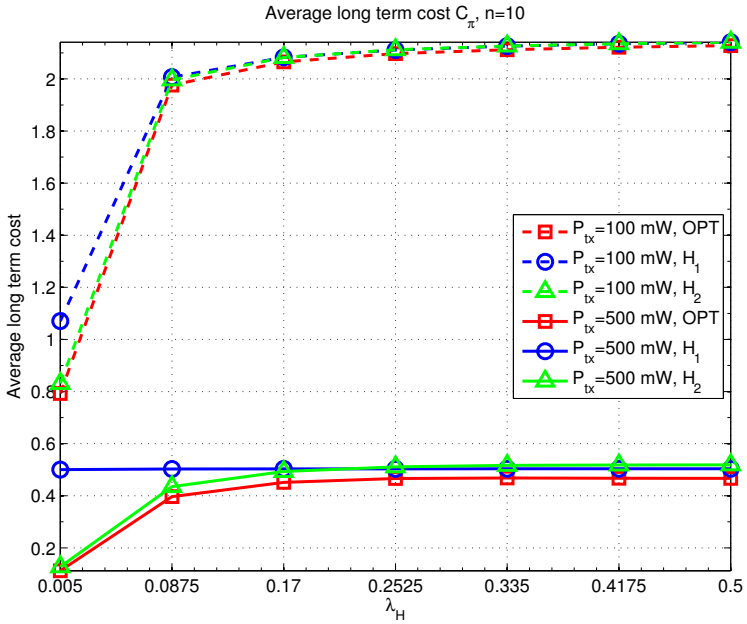

Figure 3.6: Average long term cost for $n = 10$.

Furthermore, we compare the quantitative results for $P_{tx} = 100$ mW (plotted in dashed line) and $P_{tx} = 500$ mW (plotted in solid line).

The optimal policy $\pi^*_{k+1}(i)$ $\forall i \in \mathcal{X}$ outperforms the heuristic policies H_1 and H_2 in terms of average long term cost $\forall P_{tx} \in \{100, 500\}$ mW and $\forall \lambda_H \in [0.005, 0.5]$ new code-vectors arrivals/s, as expected. Also, the optimal policy $\pi^*_{k+1}(i)$ $\forall i \in \mathcal{X}$ for $P_{tx} = 500$ mW outperforms the optimal policy $\pi^*_{k+1}(i)$ $\forall i \in \mathcal{X}$ for $P_{tx} = 100$ mW in terms of average long term cost $\forall \lambda_H \in [0.005, 0.5]$ new code-vectors arrivals/s, as expected. Nevertheless, this result could not hold for the suboptimal policies, according to the basic fact that the heuristic policies H_1 and H_2 transmit without looking at the evolution of the multipath Rayleigh fading channel throught a $(C + 1)$-state Markov Chain, i.e., without discriminating between the different settings of the channel model, with two further important considerations. First, note that the heuristic policy H_2 provides a good approximation to the optimal policy $\pi^*_{k+1}(i)$ $\forall i \in \mathcal{X}$ in terms of average long term cost $\forall P_{tx} \in \{100, 500\}$ mW and $\forall \lambda_H \in [0.005, 0.5]$ new code-vectors arrivals/s. In particular, the smaller the value of $\lambda_H \in [0.005, 0.5]$ new code-vectors arrivals/s, the closer the approximation of the heuristic policy H_2 to the optimal policy $\pi^*_{k+1}(i)$ $\forall i \in \mathcal{X}$ in terms of average long term cost, with a minimum gap in $\lambda_H = 0.005$ new code-vectors arrivals/s of 0.0113 for $P_{tx} = 500$ mW and $n = 1$ time unit. Second, note that the heuristic policy H_1 tends to provide the best approximation to the optimal policy $\pi^*_{k+1}(i)$ $\forall i \in \mathcal{X}$ in terms of average long term cost as the value of n increases from $n = 1$ time unit to $n = 10$ time units. In particular, the larger the value of $\lambda_H \in [0.005, 0.5]$ new code-vectors arrivals/s, the closer the approximation of the heuristic policy H_1 to the optimal policy $\pi^*_{k+1}(i)$ $\forall i \in \mathcal{X}$ in terms of average long term cost, with a minimum gap in $\lambda_H = 0.5$ new code-vectors arrivals/s of 0.0134 for $P_{tx} = 100$ mW and $n = 10$ time units.

The numerical results highlight that both the heuristic policies H_1 and H_2 can be exploited to replace the optimal policy $\pi^*_{k+1}(i)$ $\forall i \in \mathcal{X}$ and to suit our model formulation in some appropriate situations, i.e., when the the scheduling of a Single-Server Multiple-Buffer queueing system satisfies the application requirements and permits a good (even if suboptimal) application performance in terms of throughput and data validity, reliability, and accuracy.

3.2.2 The percentage of discarded codebook indexes

If $update(k) = 1$, then the new code-vectors arrival process is turned off until the new code-vector transmission in completed. If $update(k) = 1$ and $u_k \neq HP$ is chosen, then $update(k+1) = 1$ and $d(k+1) = [d(k)+1]^-$ w.p. 1. If $update(k) = 1$ and $u_k = HP$, then $update(k+1) = 1$ and $d(k+1) = [d(k)+1]^-$ w.p. $P^{(n)}_{KO,ij}$ or $update(k+1) = 0$ and $d(k+1) = 0$ w.p. $P^{(n)}_{OK,ij}$. This is not the case for the codebook indexes arrival process.

The percentage of discarded codebook indexes (both potential future *good codebook indexes* and potential future *bad codebook indexes*) due to the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$ is computed via simulation from index $k = 0$ to index $k = 10^6$ as the ratio between the overall number of codebook indexes that are dropped from LPQ because of the finite capacities of the buffers of the parallel queueing system and the overall number of codebook indexes arrivals.

In Figure 3.7 and Figure 3.8, we represent the percentage of discarded codebook indexes due to the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$ and due to the heuristic policies H_1 and H_2 as a function of $\lambda_H \in [0.005, 0.5]$ new code-vectors arrivals/s. Again, the red squared markers stand for the optiml policy $\pi^*_{k+1}(i)$ $\forall i \in \mathcal{X}$, the blue circular markers stand for the heuristic policy H_1, and the green upward-pointing triangular markers stand the heuristic policy H_2. In particular, Figure 3.7 reports the percentage of discarded codebook indexes due to the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$ and due to the heuristic policies H_1 and H_2 when $n = 1$ time unit, whereas Figure 3.8 reports the percentage of discarded codebook indexes due to the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$ and due to the heuristic policies H_1 and H_2 when $n = 10$ time units. Furthermore, we compare the quantitative results for $P_{tx} = 100$ mW (plotted in dashed line) and $P_{tx} = 500$ mW (plotted in solid line).

The percentage of discarded codebook indexes due to the heuristic policy H_1 is almost equal to 1 $\forall P_{tx} \in \{100, 500\}$ mW and $\forall \lambda_H \in [0.005, 0.5]$ new code-vectors arrivals/s, as expected. This is due to the basic fact that the heuristic policy H_1 is interested in the new code-vector synchronization, without taking into account the state of LPQ. It is quite interesting to note that, besides providing a good approximating to the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$ in terms

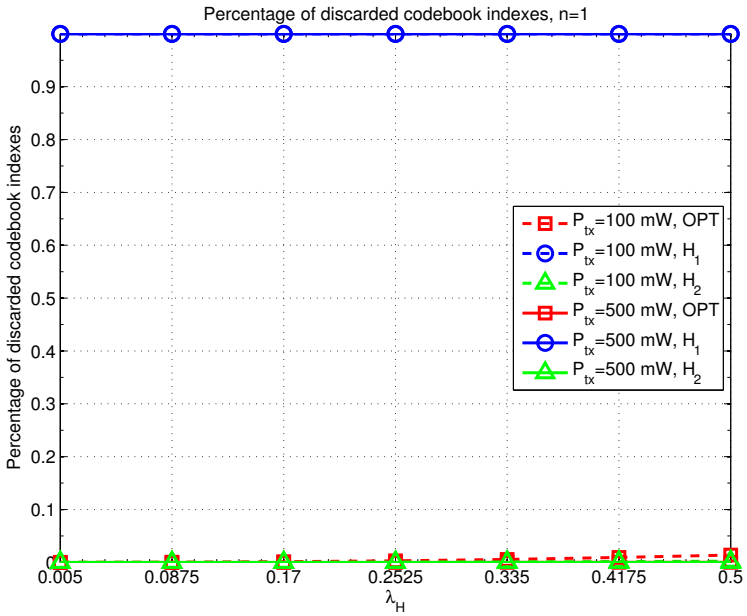

Figure 3.7: Percentage of discarded codebook indexes for $n = 1$.

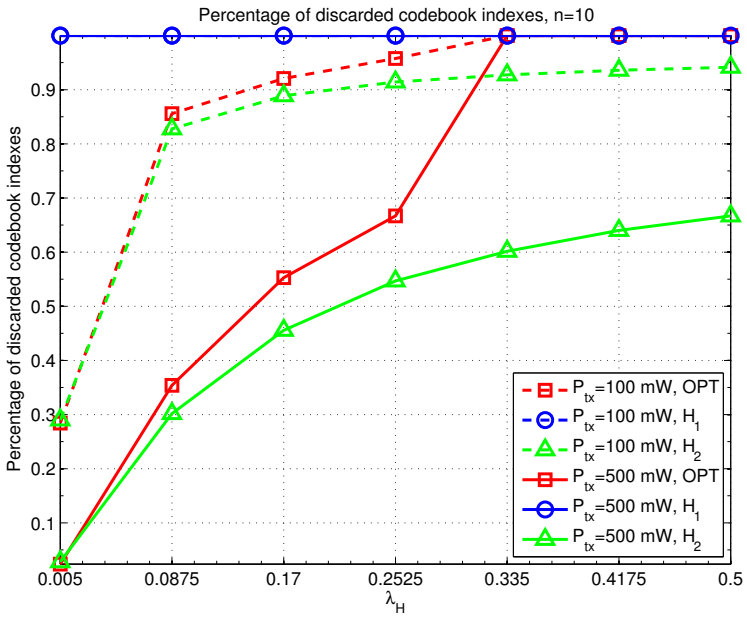

Figure 3.8: Percentage of discarded codebook indexes for $n = 10$.

of average long term cost $\forall P_{tx} \in \{100, 500\}$ mW and $\forall \lambda_H \in [0.005, 0.5]$ new code-vectors arrivals/s, the heuristic policy H_2 outperforms the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$ in terms of percentage of discarded indexes. These considerations hold even more as the value of n increases from $n = 1$ time unit to $n = 10$ time units, with a maximum gap in $\lambda_H = 0.335$ new code-vectors arrivals/s of 0.3983 for $P_{tx} = 500$ mW and $n = 10$ time units. This is due to the basic fact that the heuristic policy H_2 is not interested in the new code-vector synchronization but in throughput, without taking into account the state of HPQ. As a consequence, the heuristic policy H_2 aims at emptying LPQ, thus keeping the parallel queueing system stable.

3.2.3 The energy efficiency of $u_k = HP$ and $u_k = LP$

The energy efficiency of the control or decision variable $u_k = HP$ due to the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$ is computed via simulation from index $k = 0$ to index $k = 10^6$ as the ratio between the overall number of times in which the trasmission of the new code-vectors is successful and the overall number of times in which the control or decision variable $u_k = HP$ is chosen.

The energy efficiency of the control or decision variable $u_k = LP$ due to the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$ is computed via simulation from index $k = 0$ to index $k = 10^6$ as the ratio between the overall number of times in which the trasmission of the codebook indexes (both *good codebook indexes* and *bad codebook indexes*) is successful and the overall number of times in which the control or decision variable $u_k = LP$ is chosen.

In Figure 3.9 and Figure 3.10, we represent the energy efficiency due to the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$ and due to the heuristic policies H_1 and H_2 as a function of $\lambda_H \in [0.005, 0.5]$ new code-vectors arrivals/s. Again, the red squared markers stand for the optiml policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$, the blue circular markers stand for the heuristic policy H_1, and the green upward-pointing triangular markers stand the heuristic policy H_2. In particular, Figure 3.9 reports the energy efficiency of the control or decision variable $u_k = HP$ due to the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$ and due to the heuristic policy H_1 when $n = 10$ time units, whereas Figure 3.10 reports the energy efficiency of the

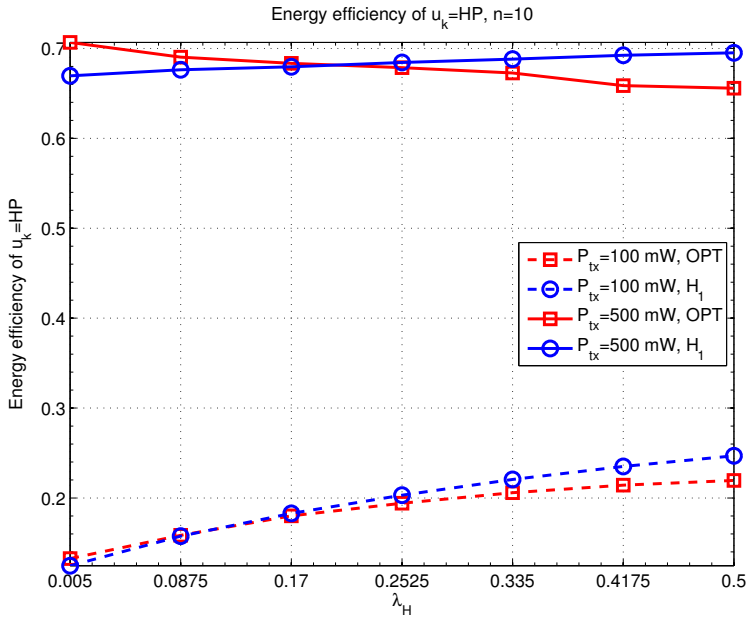

Figure 3.9: Energy efficiency of the control or decision variable $u_k = HP$.

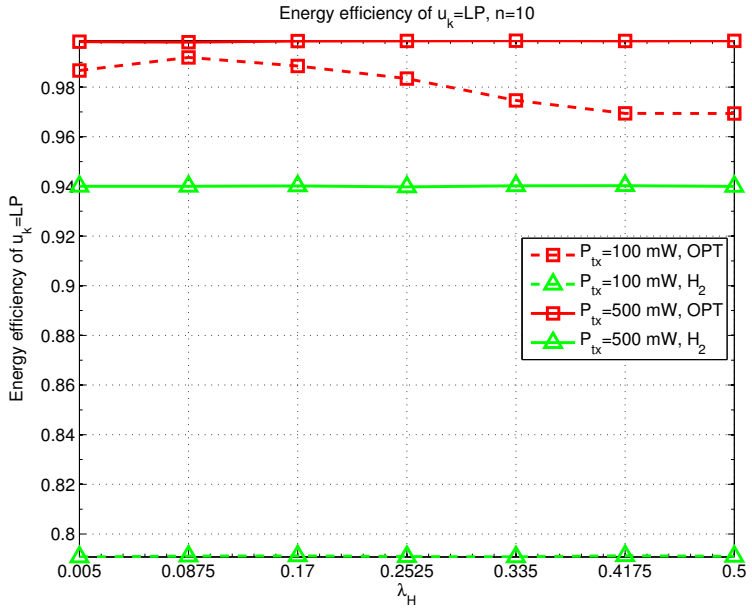

Figure 3.10: Energy efficiency of the control or decision variable $u_k = LP$.

control or decision variable $u_k = LP$ due to the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$ and due to the heuristic policy H_2 when $n = 10$ time units. Furthermore, we compare the quantitative results for $P_{tx} = 100$ mW (plotted in dashed line) and $P_{tx} = 500$ mW (plotted in solid line).

In Figure 3.9, it is quite interesting to note that the heuristic policy H_1 performs better than the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$ in terms of energy efficiency of the control or decision variable $u_k = HP$ as the value of λ_H tends to $\lambda_H = 0.5$ new code-vectors arrivals/s. Furthermore, note that there is a strong dependence of the energy efficiency of the control or decision variable $u_k = HP$ on the transmission power of the transmitting biomedical device, with an average gap of 0.4916 between $P_{tx} = 100$ mW and $P_{tx} = 500$ mW for the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$, with an average gap of 0.4878 between $P_{tx} = 100$ mW and $P_{tx} = 500$ mW for the heuristic policy H_1. In Figure 3.10, the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$ performs better than the heuristic policy H_2 in terms of energy efficiency of the control or decision variable $u_k = LP$ $\forall \lambda_H \in [0.005, 0.5]$ new code-vectors arrivals/s. In particular, the energy efficiency of the control or decision variable $u_k = LP$ due to the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$ is close to 1 $\forall P_{tx} \in \{100, 500\}$ mW and $\forall \lambda_H \in [0.005, 0.5]$ new code-vectors arrivals/s, with an average gap of 0.1895 between the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$ and the heuristic policy H_2 for $P_{tx} = 100$ mW, with an average gap of 0.0583 between the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$ and the heuristic policy H_2 for $P_{tx} = 500$ mW.

Note that the performance evaluation of the energy efficiency of $u_k = HP$ and $u_k = LP$ of the Single-Server Multiple-Buffer queueing system aims at capturing the fundamental need to avoid energy misuse. Depending on the application requiremenets, an average gap of 0.0583 between the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$ and the heuristic policy H_2 for $P_{tx} = 500$ mW can be regarded as negligible.

3.2.4 The average new code-vector delay

The average new code-vector delay due to the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$ is computed via simulation from index $k = 0$ to index $k = 10^6$ as the ratio

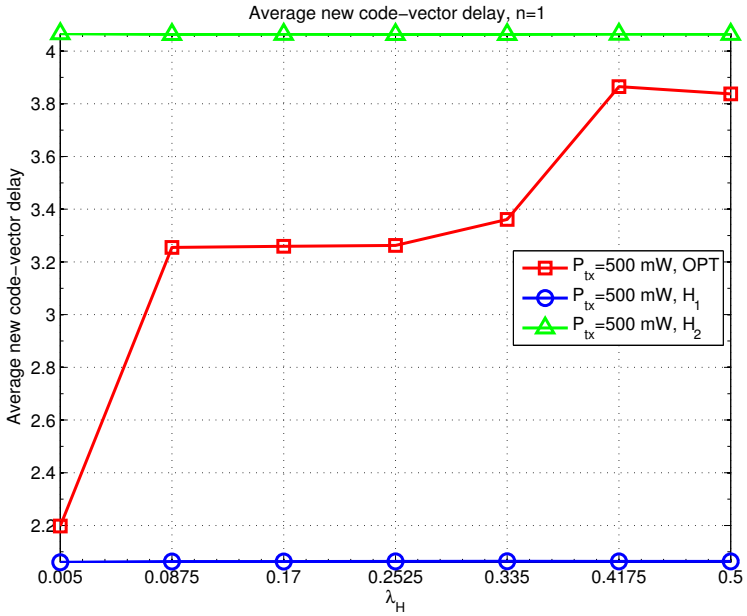

Figure 3.11: Average new code-vector delay for $n = 1$.

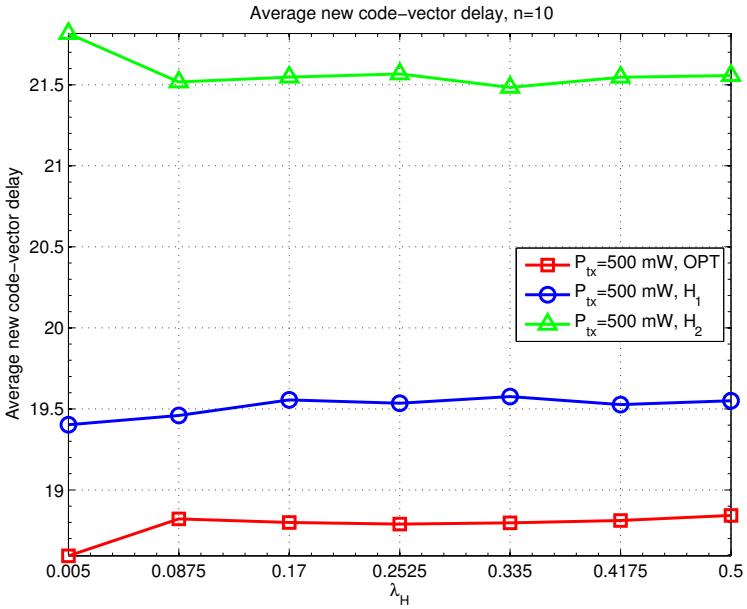

Figure 3.12: Average new code-vector delay for $n = 10$.

between the overall number of time units during which the new code-vectors are buffered in HPQ and the overall number of new code-vectors arrivals in HPQ.

In Figure 3.11 and Figure 3.12, we represent the average new code-vector delay due to the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$ and due to the heuristic policies H_1 and H_2 as a function of $\lambda_H \in [0.005, 0.5]$ new code-vectors arrivals/s. Again, the red squared markers stand for the optiml policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$, the blue circular markers stand for the heuristic policy H_1, and the green upward-pointing triangular markers stand the heuristic policy H_2. In particular, Figure 3.11 reports the average new code-vector delay due to the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$ and due to the heuristic policies H_1 and H_2 when $n = 1$ time unit, whereas Figure 3.12 reports the average new code-vector delay due to the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$ and due to the heuristic policies H_1 and H_2 when $n = 10$ time units. Furthermore, we consider $P_{tx} = 500$ mW.

The heuristic policy H_2 provides the worst performance in terms of average new code-vector delay, as expected. This is due to the basic fact that the heuristic policy H_2 is not interested in the new code-vector synchronization but in throughput, without taking into account the state of HPQ, as discussed above. As the value of n tends to $n = 1$ time unit, we observe that the average new code-vector delay due to the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$ is higher than the one due to the heuristic policy $H_1\ \forall P_{tx} \in \{100, 500\}$ mW and $\forall \lambda_H \in [0.005, 0.5]$ new code-vectors arrivals/s. As the value of n tends to $n = 10$ time units, we observe that the average new code-vector delay due to the heuristic policy H_1 is higher than the one due to the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}\ \forall P_{tx} \in \{100, 500\}$ mW and $\forall \lambda_H \in [0.005, 0.5]$ new code-vectors arrivals/s. It is quite interesting to note that the heuristic policy H_1 outperforms the heuristic policy H_2 of about 2 time units $\forall n \in \{1, 5, 10\}$ time units. This result could be better exploited in possible future developments, in order to set some lower or upper bounds to the the average new code-vector delay due to the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$.

Chapter 4

Conclusions

This thesis applies the DP and Optimal Control theory, along as the Markov Chain and Markov Decision Chain theoretical framework, to the problem of designing optimal strategies for the transmission of compressed biometric time series over wireless channels. In particular, this work focuses on the critical aspects concerning both the codebook-based approaches and the evolution of the multipath Rayleigh fading channel throught a $(C+1)$-state Markov Chain, thus exploring the issues associated with both compression and transmission. Our novel approach is in the performance evaluation of the scheduling of a Single-Server Multiple-Buffer queueing system, thus taking into account the transmission of both new code-vectors and codebook indexes. In particular, we consider different values of the transmission power of the transmitting biomedical device and different values of the new code-vectors arrival rate. Furthermore, we compare the system performance under the optimal policy, which require that the central controller has full knowledge of the system parameters and of the evolution of the multipath Rayleigh fading channel, with the system performance under two suboptimal policies (also referred to as heuristic policies H_1 and H_2), which do not require full state information and turn out to be computationally light. The numerical results highlight that both the heuristic policies H_1 and H_2 can be exploited to replace the optimal policy $\pi^*_{k+1}(i)$ $\forall i \in \mathcal{X}$ and to suit our model formulation in some appropriate situations, i.e., when the scheduling of a Single-Server Multiple-Buffer queueing system satisfies the application requirements and permits a good (even if suboptimal) application performance in terms of throughput and data validity, reliability,

and accuracy. In particular, note that there is a strong dependence of the percentage of discarded codebook indexes (both potential future *good codebook indexes* and potential future *bad codebook indexes*) on the allowed maximum number of current codebook indexes buffered in LPQ_a and on the allowed maximum number of OOS codebook indexes buffered in LPQ_b. Also, note there is a strong dependence of the average new code-vector delay on the allowed maximum number of stages during which $update(k)$ is buffered in HPQ. As a consequence, possible future developments could take into account the setting of a Single-Server Multiple-Buffer queueing system with larger values of A, B, and D, thus leading to more complex models, capable to determine how the system parameters A and B affect the metrics of interest and the behavior of the optimal policy $\pi^*_{k+1}(i)\ \forall i \in \mathcal{X}$ as a function of D. A lot of work remains to be done to couple our model formulation to the novel studies on vector quantization, pattern matching, and codebook-based approaches, with particular attention for Neural Networks. An interesting proposal for a possible future extension of this work could take into account a self-adapting codebook, thus a corresponding self-adapting Single-Server Multiple-Buffer queueing system, capable to switch between optimal and suboptimal policies, *at runtime*, according to the available resourses at the transmitting side and to the statistical description of the source.

Bibliography

[1] *http://www.ccsinsight.com*

[2] N. Bui and M. Rossi, *Staying Alive: System Design for Self-Sufficient Sensor Networks*, ACM Transactions on Sensor Networks (TOSN) 11.3 (2015): 40.

[3] N. Bui and M. Rossi, *Dimensioning Self-sufficient Networks of Energy Harvesting Embedded Devices*, Wireless Access Flexibility, Springer, pp. 138-150, 2013.

[4] J. Yang and S. Ulukus, *Optimal Packet Scheduling in an Energy Harvesting Communication System*, IEEE Transactions on Communications Volume 60 Issue 1 (2012), pp. 220-230.

[5] D. Zordan, B. Martinez, I. Vilajosana, and M. Rossi, *On the Performance of Lossy Compression Schemes for Energy Constrained Sensor Networking,*, ACM Transactions on Sensor Networks Volume 11 Issue 1 (2014), pp. 15:1-15:34.

[6] P. Castiglione, O. Simeone, E. Erkip, and T. Zemen, *Energy Management Policies for Energy-Neutral Source-Channel Coding*, IEEE Transactions on Communications Volume 60 Issue 9 (2012), pp. 2668-2678.

[7] P. Castiglione and G. Matz, *Energy-Neutral Source-Channel Coding with Battery and Memory Size Constraints*, IEEE Transactions on Communications Volume 62 Issue 4 (2014), pp. 1373-1381.

[8] X.C. Qu and Y. Zhang, *A Compression Algorithm for ECG Data Using Variable-Length Classified Template Sets*, IEEE International Symposium on Computer, Consumer and Control (IS3C) (2014), pp. 856-859.

[9] Y. Noh and D. Jeong, *Implementation of ECG Template Matching Compression Algorithm for Mobile Healthcare Device*, Journal of Information Processing and Management Volume 4 Issue 3 (2013), pp. 260-268.

[10] C.C. Sun and S.C. Tai, *Beat-Based ECG Compression Using Gain-Shape Vector Quantization*, IEEE Transactions on Biomedical Engineering Volume 52 Issue 11 (2005), pp. 1882-1888.

[11] W. Khalifa, A. Salem, M. Roushdy, and K. Revett, *A Survey of EEG Based User Authentication Schemes*, Proceedings of the 8th International Conference on Informatics and Systems (INFOS) (2012), pp. 55-60.

[12] F. Beritelli and S. Serrano, *Biometric Identification Based on Frequency Analysis of Cardiac Sounds*, IEEE Transactions on Information Forensics and Security Volume 2 Issue 3 (2007), pp. 596-604.

[13] Y. Y. Gu, Y. Zhang, and Y. Zhang, *A Novel Biometric Approach in Human Verification by Photoplethysmographic Signals*, Proceedings of the 4th IEEE Annual International Conference of the Engineering in Medicine and Biology Society (EMBS) (2003), pp. 13-14.

[14] J. Cox, H. Fozzard, F. M. Nolle, and G. Oliver, *AZTEC: A Preprocessing System for Real-Time ECG Rhythm Analysis*, IEEE Transactions on Biomedical Engineering Volume 37 Issue 9 (1968), pp. 128-129.

[15] J. P. Abenstein and W. J. Tompkins, *A New Data-Reduction Algorithm for Real-Time ECG Analysis*, IEEE Transactions on Biomedical Engineering Volume 29 Issue 1 (1982), pp. 43-48.

[16] T. Schoellhammer, B. Greenstein, M. W. E. Osterweil, and D. Estrin, *Lightweight Temporal Compression of Microclimate Datasets*, IEEE International Conference on Local Computer Networks (LCN), Tampa, FL, US, 2004.

[17] R. Shankara and S. M. Ivaturi, *ECG Data Compression Using Fourier Descriptors*, IEEE Transactions on Biomedical Engineering Volume 33 Issue 4 (1986), pp. 428-434.

[18] H. Lee and K. M. Buckley, *ECG Data Compression Using Cut and Align Beats Approach and 2-D Transforms*, IEEE Transactions on Biomedical Engineering Volume 46 Issue 5 (1999), pp. 556-564.

[19] B. A. Rajoub, *An Efficient Coding Algorithm for the Compression of ECG Signals Using the Wavelet Transform*, IEEE Transactions on Biomedical Engineering Volume 49 Issue 4 (2002), pp. 355-362.

[20] N. Maglaveras, T. Stampkopoulos, K. Diamantaras, C. Pappas, and M. Strintzis, *ECG Pattern Recognition and Classification Using Nonlinear Transformations and Neural Networks: A Review*, International Journal of Medical Informatics Volume 52 Issue 1-3 (1998), pp. 191-208.

[21] L. He, W. Hou, X. Zhen, and C.Peng, *Recognition of ECG Patterns Using Artificial Neural Network*, International Conference on Intelligent Systems Design and Applications Volume 2 (ISDA) (2006), pp.477-481.

[22] B.S. Vedavathi, S. Biradar, S.G. Hiremath, and G. Thippeswamy, *Wavelet Transform Based Neural Network Model to Detect and Characterise ECG and EEG Signals Simultaneously*, IEEE International Advance Computing Conference (IACC) (2015), pp.743-748.

[23] R. Francescon, M. Hooshmand, M. Gadaleta, E. Grisan, S. K. Yoon, and M. Rossi, *Toward Lightweight Biometric Signal Processing for Wearable Devices*, IEEE Engineering in Medicine and Biology Society (EMBS), August 25-29, Milan, Italy, 2015.

[24] A. Ramakrishnan and S. Saha, *ECG Compression by Multirate Processing of Beats*, Computers and biomedical research Volume 29 Issue 5 (1996), pp. 407-417.

[25] S.G. Miaou and J.H. Larn, *Adaptive Vector Quantisation for Electrocardiogram Signal Compression Using Overlapped and Linearly Shifted Codevectors*, Medical and Biological Engineering and Computing Volume 38 Issue 5 (2000), pp. 547-552.

[26] A. Gersho and R.M. Gray, *Vector Quantization and Signal Compression*, Kluwer Academic Publishers (1991).

[27] A. Gersho and V. Cuperman, *Vector Quantization: A Pattern-Matching Technique for Speech Coding*, IEEE Communications Magazine Volume 21 Issue 9 (1983), pp. 15-21.

[28] R.M. Gray, *Vector Quantization*, IEEE ASSP Magazine Volume 1 (1984), pp. 4-29 (Reprinted in Vector Quantization, edited by H. Abut, IEEE Press (1990)).

[29] S.P. Lloyd, *Least Squares Quantization in PCM*, IEEE Transactions on Information Theory Volume 28 Issue 2 (1982), pp.129-137.

[30] Y. Linde, A. Buzo, and R.M. Gray, *An Algorithm for Vector Quantizer Design*, IEEE Transactions on Communications Volume 8 Issue 1 (1980), pp. 84-95.

[31] W.H. Equitz, *A new Vector Quantization Clustering Algorithm*, IEEE Transactions on Acoustics, Speech, and Signal Processing, Volume 37 Issue 10 (1989), pp. 1568-1575.

[32] S.H. Huang and S.H. Chen, *Fast Encoding Algorithm for VQ-based Image Coding*, Electronics Letters Volume 26 Issue 19 (1990), pp. 1618-1619.

[33] S.W. Ra and J.K. Kim, *A Fast Mean-Distance-Ordered Partial Codebook Search Algorithm for Image Vector Quantization*, IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing Volume 40 Issue 9 (1993), pp. 576-579.

[34] L. Torres and J. Huguet, *An Improvement on Codebook Search for Vector Quantization*, IEEE Transactions on Communications Volume 42 Issue 2 (1994), pp. 208-210.

[35] D.P. Bertsekas, *Dynamic Programming and Optimal Control*, Athena Scientific, 2012.

[36] L. I. Sennott, *Stochastic Dynamic Programming and the Control of Queueing Systems*, John Wiley & Sons, 2009.

[37] E. Altman, *Constrained Markov Decision Processes: Stochastic Modeling*, London: Chapman and Hall CRC, 1999.

[38] H.S. Wang and N. Moayeri, *Finite-State Markov Channel - A Useful Model for Radio Communication Channels*, IEEE Transactions on Vehicular Technology Volume 44 Issue 1 (1995), pp. 163-171.

[39] M. Rossi, L. Badia, and M. Zorzi, *SR ARQ Delay Statistics on N-State Markov Channels with Non-Instantaneous Feedback*, IEEE Transactions on Wireless Communications Volume 5 Issue 6 (2006), pp. 1526-1536.

[40] M. Rossi, L. Badia, and M. Zorzi, *SR-ARQ Delay Statistics on N-State Markov Channels with Finite Round Trip Delay*, IEEE Global Telecommunications Conference GLOBECOM '04 Volume 5 (2004), pp. 3032-3036.