# Scheduling of Users with Markovian Time-Varying Transmission Rates[*]

Fabio Cecchi[†]
University of Pisa, Italy
BCAM — Basque Center for Applied Mathematics, Spain
cecchi@mail.dm.unipi.it

Peter Jacko[‡]
Lancaster University, UK
BCAM — Basque Center for Applied Mathematics, Spain
p.jacko@lancaster.ac.uk

## ABSTRACT

We address the problem of developing a well-performing and implementable scheduler of users with wireless connection to the base station. The main feature of such real-life systems is that the quality conditions of the user channels are time-varying, which turn into the time-varying transmission rate due to different modulation and coding schemes. We assume that this phenomenon follows a Markovian law and most of the discussion is dedicated to the case of three quality conditions of each user, for which we characterize an optimal index policy and show that threshold policies (of giving higher priority to users with higher transmission rate) are not necessarily optimal. For the general case of arbitrary number of quality conditions we design a scheduler and propose its two practical approximations, and illustrate the performance of the proposed index-based schedulers and existing alternatives in a variety of simulation scenarios.

## Categories and Subject Descriptors

F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems—*Sequencing and scheduling*; C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless communication*; C.4 [**Performance of Systems**]: Modeling techniques; G.3 [**Probability and Statistics**]: Queueing theory

## General Terms

Theory, Algorithms, Performance

## Keywords

wireless networks, opportunistic scheduling, performance evaluation, Markov decision processes, stochastic scheduling, stability

[*]The authors are grateful to U. Ayesta for many fruitful discussions and the four anonymous referees for valuable suggestions that helped to improve the presentation of the paper. This research was mostly done during the stay of F. Cecchi at BCAM during August-October 2012 funded by the BCAM Internship program. This research was partially supported by grant MTM2010-17405 (Ministerio de Ciencia e Innovación, Spain).

[†]Full addresses: Dipartimento di Matematica, University of Pisa, Largo Bruno Pontecorvo 5, 56127, Pisa, Italy; BCAM - Basque Center for Applied Mathematics, Mazarredo 14, 48009 Bilbao (Basque Country), Spain.

[‡]P. Jacko is co-funded by the LANCS Initiative. Full addresses: Department of Management Science, Lancaster University Management School, Lancaster, LA1 4YX, UK; BCAM - Basque Center for Applied Mathematics, Mazarredo 14, 48009 Bilbao (Basque Country), Spain.

## 1. INTRODUCTION

This paper is motivated by the necessity of designing an appropriate scheduler for wireless systems such as Long Term Evolution (4G LTE), heterogeneous networks (HetNet) or vehicular communications systems. Such a scheduler must be capable of exploiting the base station's capacity to serve the heterogeneous demands of the users that are within the base station's power range in order to optimize the system performance and user experience. We model such a system as the multi-class queueing system with multiple preemptive servers, in which users of different classes randomly arrive and depart once their job is completed. Different classes may have different sets of accessible transmission rates associated with the finite class-dependent number of quality conditions of the channels, whose evolution is class-dependent and Markovian. Further, the classes may have heterogeneous waiting costs and mean job sizes. The model covers both the downlink and synchronized uplink wireless systems.

Several schedulers have been proposed recently for such a *flow-level* scheduling problem based on ad-hoc arguments, simulation outcomes or approximate optimization, e.g., in [17, 9, 8, 1, 3, 15]. The pioneering work was done by [17], who showed that the system capacity can be improved by opportunistically serving users whose current transmission rate is maximal. Such a scheduler, known as the *MaxRate* scheduler in the wireless networks literature is thus *naively opportunistic*: it is myopically throughput optimal (maximizing one-slot transmission rate) and simple to implement, but it ignores the possible future evolution and was shown to perform bad in the long-term. For instance, it may quickly become unstable (i.e., the number of waiting users explodes) as the load increases, while other schedulers may keep the system stable [1, 3]. It may also be extremely unfair to users whose highest accessible transmission rate is lower than the transmission rates of others. This scheduler is also known as the $c\mu$-rule in the stochastic scheduling literature, and we will adopt this name in this paper.

Gradient-based schedulers are the state-of-the-art in opportunistic scheduling, in particular the Proportionally Fair (PF) scheduler, patented [11], was proposed to be implemented in the CDMA 1xEV-DO system of 3G cellular networks [6]. PF maximizes the logarithmic throughput of the network, providing thus an improved fairness over $c\mu$-rule [18]. [9] analyzed flow-level stability of PF by approximating it by the *Relatively Best* (RB) scheduler, which gives priority to users according to their ratio of the current transmission rate to the mean transmission rate. This scheduler is thus *fairly opportunistic*: it takes the possible future evolution into account, it is not myopically throughput optimal, performs well with respect to guaranteeing a minimal throughput to the users with low accessible transmission rates, however, it is not maximally stable at flow-level [1].

The schedulers proposed in [8, 1, 3], called the *Score Based* (SB) [7], *Proportionally Best* (PB) and *Potential Improvement* (PI) [5] schedulers, respectively, belong to the family of the *best-condition* schedulers. A best-condition scheduler gives absolute priority to the users in their respective best accessible quality condition over the others, hence ignoring the transmission rate associated with such a best condition. Such a policy is thus *smartly opportunistic*: this feature still ignores the possible future evolution, but it is not myopic, and turns out to perform well in the long-term and in heavily loaded systems, for being maximally stable [4, 16]. Fairness of best-condition schedulers has not been addressed adequately yet, only [3] illustrated in one simulation scenario that PI maintained the average number of uncompleted jobs significantly more balanced than other schedulers.

It has been typically assumed in the previous work that the channel evolution is independent and identically distributed (iid). Consideration of Markovian channel evolution (rather than the iid evolution) is however important, because it is known that channels do have a memory, although the precise evolution is usually unknown or difficult to estimate (and moreover can change over time). For instance, random processes in signal processing are often modeled by the autoregressive model of order 1, which is Markovian (and not iid).

In this paper we give insights into answers to the fundamental questions about scheduling in the Markovian channels setting:

- What is the structure of an optimal policy?
- Why are there no optimality results available? (Only maximal stability has been established.)
- How do the actions taken in the non-best states influence the performance? (Maximal stability only indicates what to do in the channel condition with the highest transmission rate.)
- (When) are the maximally stable policies preferable in practice?
- How to resolve the trade-off between being naively opportunistic, smartly opportunistic, and prioritizing "short" jobs?
- Do the maximally stable policies perform well even in case of classes with heterogeneous waiting costs?
- How fair are the maximally stable policies?

We are still far from providing definite answers, but we believe that this paper can develop some intuition and point to the main issues and avenues to focus on in future research in this area.

In section 2 we formalize the system and the scheduling problem. An MDP approach is described in section 3 in order to formulate a single-user optimization problem in which a price must be paid for service. This problem is addressed in section 4, where we develop index policies and study solvability by threshold policies. We solve the problem for channel evolution over three quality conditions and partially characterize the optimal solution in general. It is important to note that threshold policies (of giving higher priority to users with higher transmission rate) are not necessarily optimal. That is, when a single wireless user is competing even with a non-time-varying user, it may be optimal to serve the wireless user in channel conditions with the highest and the lowest transmission rates, but to prefer the other user if the wireless one is in channel condition with the medium transmission rate. Based on these results, we propose a new scheduler and two practical approximations in section 5. Their performance is evaluated and contrasted with schedulers proposed in previous literature in section 6. Finally, section 7 concludes. The proofs are omitted due to lack of space.

## 2. PROBLEM DESCRIPTION

We consider a time-slotted system, so that we study a discrete-time job scheduling problem. The decisions are taken in time epochs/instants $t \in \mathcal{T} := \{0, 1, \dots\}$, and are applied during the time slots $t \in \mathcal{T}$ of duration $\tau$ seconds, where slot $t$ corresponds to the interval between epochs $[t, t+1)$.

### 2.1 Job-Channel-User Classes

Suppose that there are $K$ classes of users, labeled $k \in \mathcal{K} := \{1, 2, \dots, K\}$. Each user is uniquely associated with the job it requests to download and with the dedicated wireless channel.

#### *User Arrivals.*

For each class $k \in \mathcal{K}$, the number of class-$k$ users arriving to the system, $A_k(t)$, at each time epoch $t \in \mathcal{T}$, creates an iid arrival process $\{A_k(t)\}_{t \in \mathcal{T}}$ with generic element $A_k$ and mean $\lambda_k := \mathbb{E}_0[A_k] < \infty$, where $\mathbb{E}_0[\cdot]$ denotes the expectation conditional to information available at time epoch 0. The arrivals are assumed to be mutually independent across user classes.

#### *Job Sizes.*

The (integer-valued) job/flow size $b_k$ of class-$k$ user is measured in *bits* and has the geometric distribution with $\mathbb{E}[b_k] < \infty$ for classes $k \in \mathcal{K}$. This assumption is the main limitation of existing models (including this paper), but to the best of our knowledge there has not been any attempt to analytically approach the case of non-geometric job sizes in the literature.

#### *Channel Conditions.*

For each user, the quality of the channel (the *channel condition*) is changing from slot to slot, independently of all other users present in the system (including other users of the same class) and, for each class-$k$ user, takes values in the finite set $\mathcal{N}'_k := \{1, 2, \ldots, N_k\}$. Moreover, the channel condition an arriving class-$k$ user finds the channel in, is also independent of channel conditions of other users and the slot it arrives at, and is $n$ with probability $q_{k,n} \geq 0$, which satisfies $\sum_{n \in \mathcal{N}'_k} q_{k,n} = 1$.

*Channel Condition Transitions.*

We assume that at each slot, for a class-$k$ user, its channel condition evolves according to a distribution which may depend on $k$ and on the channel condition in the current slot (i.e., is Markovian). Thus, for each user of class $k \in \mathcal{K}$, we can define a (time-homogeneous) Markov chain with state space $\mathcal{N}^{(k)}$.

We denote by $q_{k,n,m} := \mathbb{P}(Z_k(t+1) = m | Z_k(t) = n)$ the probability that the channel quality condition of a class-$k$ user moves from the state $n$ to the state $m$ in one slot. The class $k$ channel condition transition probability matrix is thus

$$\boldsymbol{Q}_k := \begin{array}{c} \\ 1 \\ 2 \\ \vdots \\ N_k \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & \ldots & N_k \end{array} \\ \begin{pmatrix} q_{k,1,1} & q_{k,1,2} & \cdots & q_{k,1,N_k} \\ q_{k,2,1} & q_{k,2,2} & \cdots & q_{k,2,N_k} \\ \vdots & \vdots & \ddots & \vdots \\ q_{k,N_k,1} & q_{k,N_k,2} & \cdots & q_{k,N_k,N_k} \end{pmatrix} \end{array}$$

where $\sum_{m \in \mathcal{N}'_k} q_{k,n,m} = 1$ for every condition $n \in \mathcal{N}'_k$. We emphasize that channel condition transitions of users are independent across users. In this paper we assume that $\boldsymbol{Q}_k$ is irreducible and aperiodic for every $k$.

*Transmission Rates.*

Different channel conditions correspond to different transmission rates associated with the available modulation and coding schemes (MCS). When a class-$k$ user is in channel condition $n \in \mathcal{N}'_k$, she can receive data at transmission rate $s_{k,n}$ bits per second. Without loss of generality we assume that the higher the channel condition, the higher the transmission rate, i.e., $0 \leq s_{k,1} < s_{k,2} < \cdots < s_{k,N_k}$.

To avoid trivial cases, we assume that each class $k$ can be served, i.e., $s_{k,N_k} > 0$. We further restrict our attention to the case in which at least one class $k$ is time-varying, i.e., $s_{k,1} < s_{k,N_k}$.

For each class $k$ we define $B_k := \lceil b_k / s_{k,N_k} \rceil$, the number of slots in the best quality condition ($N_k$) needed to complete the job. Thus, $B_k$ is the minimum number of slots that a job of size $b_k$ must spend in service in order to be completed. We denote by $\mathbb{E}[B_k]$ (positive integer) the mean of this random variable of class $k$. We further define the traffic intensity of class $k$ as $\varrho_k := \lambda_k \mathbb{E}[B_k]$.

*Waiting Costs.*

For every user of class $k$ the operator accrues a waiting cost $c_k > 0$ for every slot while it is uncompleted.

## 2.2 Server

At the beginning of every slot $t$, the server (base station) observes the actual state of all the users present in the system, and decides which (up to its capacity constraint $C$) of them to serve during the slot. We assume that the server is preemptive, that is at every decision epoch it is permitted to suspend the service of a user whose job is not yet concluded. Moreover, the server is allowed to be allocated to an already completed job, in this case no transmission occurs. Motivated by practical implementation, the observations of the processes defined below at epoch $t$ always include arrivals at epoch $t$, while during the time interval $(t, t+1)$ only service but no new arrivals occur.

## 2.3 Objectives

The aim is to identify scheduling policies that perform well with respect to the following objectives (or their combination):

- minimization of the expected time-average waiting cost per user;
- minimization of the expected time-average number of uncompleted jobs per slot;
- maximization of some time-average fairness function across classes.

## 3. MDP APPROACH

In this section we set out to employ a Markov decision process (MDP) approach to design a well-grounded scheduling policy. Indeed, we extend the modeling framework introduced for the scheduling problem with iid channel condition evolution in [3] based on restless bandits [21]. Other scheduling policies were designed in an ad hoc way ([8, 1]), or based on solving an optimization problem under the time-scale separation assumption ([2]).

In order to employ the MDP framework, we simplify the problem: we assume a fixed population of users present in the system at the initial slot (i.e., ignore the arrivals). That is, there is a single user of each class $k$. On the other hand, in order to admit an analytical approach, we introduce discounting of the waiting costs, with discount factor $0 \leq \beta < 1$. The results for the undiscounted case, which corresponds to the time-average criterion, will be obtained in the limit $\beta \to 1$.

These twists are still not sufficient to solve the problem optimally. Nevertheless, this approach was shown useful for designing in the iid special case a scheduler [3] that is well-performing, maximally stable and fluid-optimal under arbitrary arrivals [4]. The approach requires to analyze a single-user problem in which one pays price $\nu$ for service. This parameter appears from the Lagrangian relaxation (omitted here) as the Lagrangian parameter.

Before defining the MDP elements, we will further need to define the departure probabilities. We denote by $\mu_{k,n}$ the probability that the job $k$ in state $n$, if served, will be completed in the current slot. Since we consider jobs with geometric size, we can employ the results from [3, 15] that $\mu_{k,n} = \min\{1, 1 - (1 - 1/\mathbb{E}_0[b_k])^{\tau s_{k,n}}\}$ which can be approximated, if $\tau s_{k,n} / \mathbb{E}_0[b_k] \approx 0$ by

$$\mu_{k,n} \approx \tau s_{k,n} / \mathbb{E}_0[b_k]. \tag{1}$$

Note that the departure probabilities are increasing: $0 \leq \mu_{k,1} < \cdots < \mu_{k,N_k} \leq 1$.

### 3.1 Job-Channel-User MDP

At the beginning of every time slot, the generic user $k$ can be allocated zero capacity of the base station or be one of the users served. We denote by $\mathcal{A}_k$ the action space relative to user $k$. We have that $\mathcal{A}_k := \{0,1\}$ where the action 0 means not serving, while action 1 means serving.

Every job-channel-user $k$ is characterized by the tuple $(\mathcal{N}_k, (\boldsymbol{W}_k^a)_{a\in\mathcal{A}}, (\boldsymbol{R}_k^a)_{a\in\mathcal{A}}, (\boldsymbol{P}_k^a)_{a\in\mathcal{A}})$, where

- $\mathcal{N}_k := \{0\} \cup \mathcal{N}_k'$ is the *state space*, the state 0 indicates that the job is completed, while the set $\mathcal{N}_k'$ represents the possible channel quality conditions;
- $\boldsymbol{W}_k^a := (W_{k,n}^a)_{n\in\mathcal{N}_k}$, where $W_{k,n}^a$ is the expected one-slot capacity consumption, or *work* required by user $k$ at state $n$ if action $a$ is selected at a time epoch. Specifically, for every state $n \in \mathcal{N}_k$,
$$W_{k,n}^1 := 1, \qquad W_{k,n}^0 := 0;$$
- $\boldsymbol{R}_k^a := (R_{k,n}^a)_{n\in\mathcal{N}_k}$, where $R_{k,n}^a$ is the expected one-slot *reward* earned by user $k$ at state $n$ if action $a$ is selected at a time epoch. Specifically, for every state $n \in \mathcal{N}_k'$,
$$R_{k,0}^a := 0, \quad R_{k,n}^1 := -c_k(1-\mu_{k,n}), \quad R_{k,n}^0 := -c_k;$$
- $\boldsymbol{P}_k^a := (p_{k,n,m}^a)_{n,m\in\mathcal{N}_k}$, where $p_{k,n,m}^a$ is the probability for user $k$ of moving from state $n$ to state $m$ if action $a$ is chosen at a time epoch. These one-slot *state-transition probability matrices* are
$$\boldsymbol{P}_k^0 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & q_{k,1,1} & \cdots & q_{k,1,N_k} \\ 0 & q_{k,2,1} & \cdots & q_{k,2,N_k} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & q_{k,N_k,1} & \cdots & q_{k,N_k,N_k} \end{pmatrix},$$
$$\boldsymbol{P}_k^1 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ \mu_{k,1} & \widetilde{\mu}_{k,1}q_{k,1,1} & \cdots & \widetilde{\mu}_{k,1}q_{k,1,N_k} \\ \mu_{k,2} & \widetilde{\mu}_{k,2}q_{k,2,1} & \cdots & \widetilde{\mu}_{k,2}q_{k,2,N_k} \\ \vdots & \vdots & \ddots & \vdots \\ \mu_{k,N_k} & \widetilde{\mu}_{k,N_k}q_{k,N_k,1} & \cdots & \widetilde{\mu}_{k,N_k}q_{k,N_k,N_k} \end{pmatrix},$$

where we have denoted by $\widetilde{\mu}_{k,n} := 1 - \mu_{k,n}$.

The dynamics of user $k$ are thus captured by the *state process* $X_k(\cdot)$ and the *action process* $a_k(\cdot)$, which correspond to state $X_k(t) \in \mathcal{N}_k$ and action $a_k(t) \in \mathcal{A}$ at all time $t \in \mathcal{T}$. At time slot $t$ the choice of action $a_k(t)$ for the user $k$ in state $X_k(t)$ entails the consumption of the allocated capacity (work), the gain of the reward and the evolution of the state to $X_k(t+1) \in \mathcal{N}_k$.

## 3.2 Optimization Problem

We present now the optimization problem we consider. Let $\Pi_{X_k,a_k}$ be the space of all randomized, history dependent and non-anticipative policies, depending on the state-process $X_k(\cdot)$ and deciding the action-process $a_k(\cdot)$.

Let $\mathbb{E}_0^\pi$ denote the expectation over the future state process $X_k(\cdot)$ and the action process $a_k(\cdot)$, conditioned on the initial state $X_k(0)$ and on the policy $\pi \in \Pi_{X_k,a_k}$. For the given discount factor $\beta$ and for every value of price $\nu$, the aim is to find a policy minimizing the expected waiting cost over an infinite horizon under the discounted criterion,

$$\max_{\pi\in\Pi_{X_k,a_k}} \sum_{t=0}^{\infty} \beta^t \, \mathbb{E}_0^\pi \left[ R_{k,X_k(t)}^{a_k(t)} - \nu W_{k,X_k(t)}^{a_k(t)} \right]. \tag{2}$$

## 4. INDEX-BASED SOLUTION

In this section we focus on the single-user subproblem (2) for a generic user $k$, and we will omit the subscript $k$ to simplify the notation.

### 4.1 Index Values and Threshold Policies

Let us adapt to our scenario the definition of *index values* and *indexability*, following [14].

DEFINITION 1 (INDEXABILITY). *We say that the problem* (2) *is* indexable *if there exist values $\nu_n^* \in \mathbb{R} \cup \{-\infty, \infty\}$ for all $n \in \mathcal{N}$ such that*

1. *it is optimal to serve the user in state $n$ if $\nu_n^* \geq \nu$, and*
2. *it is optimal not to serve the user in state $n$ if $\nu_n^* \leq \nu$.*

*Such values $\nu_n^*$ are called the (Whittle) index values, and define an optimal* index policy *for the problem.*

As described in [12, 21, 19] the optimal solution can sometimes be found by means of the index policies. The index values represent, in certain way, the benefit which is obtained by serving an user in a certain state. It has been shown that for some non-trivial problems such index may not exist.

From the point of view of intuition and implementability, one is often interested in solving the problem by threshold policies.

DEFINITION 2 (SOLVABILITY BY THRESHOLD POLICIES). *We say that the problem* (2) *is* solvable by threshold policies *if for any value of $\nu$, there exists a threshold state $n(\nu)$ such that*

1. *it is optimal to serve the user in state $n$ if $n \geq n(\nu)$, and*
2. *it is optimal not to serve the user in state $n$ if $n < n(\nu)$.*

*Such policies are called* threshold policies.

We can see that the indexability property is much more general than solvability by threshold policies. Indeed, if the problem is indexable and the index values are non-increasing in $n$, then it is solvable by threshold policies. However, an indexable problem is not be solvable by threshold policies if the index values are not non-increasing in $n$; then the optimal solution may look counter-intuitive.

The restless bandit problem and their index-based solution was introduced in [21], generalizing the so-called Gittins index policy that was proved optimal for the multi-armed bandit problem in [13]. [21] gave an intuitive definition of indices. An algorithm for computing index values and sufficient indexability conditions were introduced later; see [19] for a survey. The algorithm is called *Adaptive-Greedy*, shortly *$\mathcal{AG}$-algorithm*. It was also shown that if a problem is indexable then the $\mathcal{AG}$-algorithm computes the index values.

### 4.2 Index Values Characterization

The arguments in this section are based on the conjecture of indexability.

CONJECTURE 1. *Problem* (2) *is indexable.*

Establishing indexability by the currently known approaches is likely to be technically extremely tedious. However, we believe in its validity based on the computational testing we have performed on many (including random) problem instances. Also, indexability was proved and index values were characterized in two important special cases in [3, 15]. We state it below for completeness.

THEOREM 1 ([3]). *If the channel condition evolves in an iid fashion, i.e., $q_{n,m} = q_m$ for each $n \in \mathcal{N}'$, then problem (2) is indexable and the index values are*

$$\nu_n^* = \frac{c\mu_n}{1 - \beta + \beta \sum_{m=n+1}^{N} q_m(\mu_m - \mu_n)}.$$

THEOREM 2 ([15]). *If the channel evolves according to the Gilbert-Elliot model, i.e., $N = 2$, then problem (2) is indexable and the index values are*

$$\nu_2^* = \frac{c\mu_2}{1-\beta} \qquad \nu_1^* = \frac{c\mu_1}{1 - \beta + \beta q_{1,2}^*(\mu_2 - \mu_1)},$$

*where*

$$q_{1,2}^* = \frac{1}{\frac{\beta(1-\mu_2)}{q_2^{SS}} + \frac{1-\beta(1-\mu_2)}{q_{1,2}}}.$$

We now continue with characterization of the index values in our, general, model. The following theorem identifies the highest index value in closed form.

THEOREM 3. *Under Conjecture 1, the index value $\nu_N^* = \dfrac{c\mu_N}{1-\beta}$ and we have that $\nu_N^* \geq \nu_n^*$ for every $n \in \mathcal{N}'$.*

This result is thus an extension to the Markovian setting of the characterization of the highest index value in the iid and 2-state special cases stated above. What we see is that the highest index value is *always* associated with the state with the highest transmission rate, $N$, and, rather surprisingly, it *always* has a simple expression, which grows to $+\infty$ as $\beta \to 1$.

## 4.3 Index Values for 3-State Channel

Now we concentrate on problem (2) in the case $N = 3$. If Conjecture 1 holds, we already know by Theorem 3 that the highest index is the one associated with state 3. There are therefore two possibilities: the index value of state 2 is greater than that of state 1 (i.e., the problem is solvable by threshold policies), or vice versa.

THEOREM 4. *Under Conjecture 1, if problem (2) with $N = 3$ is solvable by threshold policies, then*

$$\nu_2^* = \frac{c\mu_2}{q_2^*(\mu_3 - \mu_2)},$$

*where*

$$q_2^* := \frac{1}{\frac{1-\mu_3}{q_3^{SS}} + \frac{\mu_3}{\overline{q}_2}}, \qquad \overline{q}_2 := q_{1,3}p_1^{(2)} + q_{2,3}p_2^{(2)},$$

*where the weights $p_1^{(2)} = \frac{q_{21}}{1-q_{11}+q_{21}}$ and $p_2^{(2)} = \frac{1-q_{11}}{1-q_{11}+q_{21}}$ are the elements of the steady state probability vector of the $2 \times 2$ matrix created from $\boldsymbol{Q}$ by omitting row 3 and merging column 3 with 2. If the problem is solvable by threshold policies after relabeling states 1 and 2, then these results hold as well.*

We can further characterize the index value of state 1. Let us denote by

$$\begin{aligned}\alpha = &-q_{2,1}q_{1,2} + q_{1,1}q_{2,2} + q_{1,2}q_{3,1} - q_{1,1}q_{3,2} + q_{2,1}q_{3,2} \\ &- q_{3,1}q_{2,2} - q_{1,1} - q_{2,2} + q_{3,1} + q_{3,2} + 1,\end{aligned}$$

$$\begin{aligned}U = &(1-\mu_2)(1-\mu_3)\alpha + \mu_3(1-\mu_2)(1 - q_{2,2} + q_{1,2}) \\ &+ \mu_2(1-\mu_3)(1 - q_{3,3} + q_{1,3}) + \mu_2\mu_3,\end{aligned}$$

$$\begin{aligned}V = &(\mu_2 - \mu_1)[(q_{1,2}q_{3,1} - q_{1,1}q_{3,2} + q_{3,2})(1-\mu_3) + q_{1,2}\mu_3] \\ &+ (\mu_3 - \mu_1)[(-q_{1,2}q_{2,1} + q_{1,1}q_{2,2} - q_{2,2} \\ &- q_{1,1} + 1)(1-\mu_2) + q_{1,3}\mu_2].\end{aligned}$$

THEOREM 5. *If Conjecture 1 holds for problem (2) with $N = 3$ in the undiscounted case ($\beta = 1$) and it is solvable by threshold policies, then the index value of state 1 is the lowest and equals*

$$\nu_1^* = \frac{c\mu_1 U}{V}. \tag{3}$$

Unfortunately, we have not been able to write this formula in a more readable form. However, an interesting approximation can be obtained for large jobs.

THEOREM 6. *Let us fix a bound $M$ such that $\mu_3 \leq M \leq 1$, i.e., in view of (1) the expected job size is approximately at least $\tau s_N / M$ bits. Then we have that the index value of state 1,*

$$\nu_1^* = \frac{c\left(\mu_1 + \mathcal{O}\left(M^2\right)\right)}{\sum_{m=2,3} q_m^{SS}(\mu_m - \mu_1) + \mathcal{O}\left(M^2\right)}, \tag{4}$$

*and of state 2,*

$$\nu_2^* = \frac{c\left(\frac{\mu_2}{q_3^{SS}} + \mathcal{O}\left(M^2\right)\right)}{\mu_3 - \mu_2}. \tag{5}$$

As a consequence, if $M$ is small enough so that terms $\mathcal{O}(M^2)$ can be neglected, then we have the following approximation for the index value of state 1,

$$\nu_1^* \approx \frac{c\mu_1}{\sum_{m=2,3} q_m^{\mathrm{SS}}(\mu_m - \mu_1)}, \tag{6}$$

and of state 2,

$$\nu_2^* \approx \frac{c\mu_2}{q_3^{\mathrm{SS}}(\mu_3 - \mu_2)}. \tag{7}$$

This characterization is nothing but the index value in the iid setting (cf. Theorem 1), where the steady-state distribution is employed while the underlying Markovian channel evolution is irrelevant. The precision of this approximation is excellent for large jobs, as showed in Table 1 for condition 1. Both the absolute error and the relative error increase approximately linearly in $M$, i.e., decrease hyperbolically in job size.

Note that the larger the job, the smaller the parameter $M$, and the precision of this approximation could be interpreted as a sort of time-scale separation effect arising naturally in the solution: the *steady-state channel distribution* approximates well in which channel conditions the job will be served, whereas for shorter jobs the Markovian channel

| $M$ | Absolute Error | Relative Error | $\varepsilon$ |
|---|---|---|---|
| 1 | 0.3880 | 14.08% | $+\infty$ |
| 0.5 | 0.1854 | 7.424% | 0.16667 |
| 0.3 | 0.1273 | 4.498% | 0.04286 |
| 0.1 | 0.0399 | 1.571% | 0.00370 |
| 0.05 | 0.0237 | 0.828% | 0.00088 |
| 0.01 | 0.0051 | 0.176% | 0.00003 |
| 0.001 | 0.0005 | 0.017% | 0.00000 |

**Table 1: Mean absolute and relative errors of the approximation (6) in a sample of 2000 job-channel-user instances for each upper bound $M \geq \mu_3$.**

evolution may be more important indicating which channel condition is hit first if starting from the current condition. However, note that this phenomenon differs from the time-scale separation as often simplistically assumed in other literature, which implies that the jobs realize the *time-average throughput*, see [2].

### 4.4 Solvability by Threshold Policies for 3-State Channel

We have given in the previous subsection formulas for computing the index values of a 3-state channel assuming they are solvable by threshold policies in the undiscounted case $\beta = 1$. We will give now two sufficient conditions for having such property satisfied and we will observe that they are satisfied in a large number of problem instances.

THEOREM 7. *If Conjecture 1 holds for problem (2) with $N = 3$ in the undiscounted case ($\beta = 1$), then we have that $q_{13} \geq q_{23}$ implies that the index value of state 2 is greater than or equal to the index value of state 1, i.e., the problem is solvable by threshold policies.*

This fact seems quite evident, indeed $q_{13} \geq q_{23}$ means that the one-slot probability to move to any better state is surely higher if the user is in state 1 than in state 2. Moreover we observe that the hypothesis is satisfied for the iid special case, recovering again the the result of solvability by threshold policies by [3].

THEOREM 8. *Let us denote by $\Delta := \min\{\mu_3 - \mu_2, \mu_2 - \mu_1\}$ and $1 > M \geq \mu_3$. If Conjecture 1 holds for problem (2) with $N = 3$ in the undiscounted case ($\beta = 1$), then we have that*

$$\Delta \geq \varepsilon := \frac{M^2}{3(1-M)}$$

*implies that the index value of state 2 is greater than or equal to the index value of state 1, i.e., the problem is solvable by threshold policies.*

In the last column of Table 1 we show how small could be $\Delta \geq \varepsilon$ given a range of upper bounds $M \geq \mu_3$. This condition seems to be really strong if $M$ is small, which is the condition we required to employ the approximation of $\nu_1^*$ in the previous subsection. We emphasize that this is still quite a rough sufficient condition (see the proof). Finally, we remark that the counter-intuitive case that the index value of state 1 is greater than the index value of state 2 happens with frequency of around 2.5% for $\beta = 0.999$.

**Algorithm 1** Algorithmic scheme of PI* scheduler

```
At every slot $t$,
$C'$ ← Number of users with uncompleted jobs in their condition $N_k$
if $C' \geq C$ then
    Serve $C$ users from among the users in their condition $N_k$ (breaking ties randomly)
else
    Serve $C'$ users in their condition $N_k$
    Serve $C - C'$ users not in condition $N_k$ with highest index value $\nu^*_{k,X_k(t)}$ (breaking ties randomly)
end if
```

## 5. PROPOSED SCHEDULERS

Now we come back to the original multi-class problem with arrivals, as described in section 2. We set out to design feasible schedulers for the problem where it is allowed to serve up to $C$ users in every slot. We are interested in the undiscounted case, which is essentially the case of optimization under the time-average criterion. We will do so by deploying the results obtained in the previous section. We thus define the *Markovian Potential Improvement* (PI*) scheduler, which is written algorithmically in Algorithm 1.

However, as we have seen in the previous section, the index values $\nu^*_{k,n}$ are likely not to admit a simple closed-form characterization in the general setting, except for $\nu^*_{k,N_k} = +\infty$. We nevertheless gave a closed-form solution for $N_k = 3$. Therefore, we propose two approximations for the index values, which give rise to additional two new schedulers for general $N_k$.

First, we define the PI$^{\mathcal{AG}}$ scheduler, which approximates $\nu^*_{k,n}$ for $n \in \mathcal{N} \setminus \{N\}$ by running the $\mathcal{AG}$-algorithm with $\beta$ as close as possible to 1 while avoiding numerical instability problems. Note that this algorithm performs $\mathcal{O}(N_k^3)$ elementary operations, and requires the knowledge of the matrix $\boldsymbol{Q}_k$. On the other hand, this algorithm is likely to identify if the threshold policies are not optimal, and so these approximated index values may not necessarily be increasing in $n$.

Second, we define the PI$^{\mathrm{SS}}$ scheduler, which approximates $\nu^*_{k,n}$ for $n \in \mathcal{N} \setminus \{N\}$ by the formula

$$\frac{c_k \mu_{k,n}}{\sum_{m>n} q^{\mathrm{SS}}_{k,m}(\mu_{k,m} - \mu_{k,n})}. \tag{8}$$

This approximation is based on conjecturing generalizability of Theorem 6, which requires that $\mu_{k,N_k} \leq M$, where $M$ is small enough so that terms bounded by $M^2$ can be neglected. It is easy to prove that these approximated index values are increasing in $n$ and that their computation requires $\mathcal{O}(N_k)$ elementary operations (once the steady-state distribution is known). On the other hand, knowledge of the matrix $\boldsymbol{Q}_k$ is not required, since only the steady-state distribution is used, which may be significantly easier and more precise to estimate in practice.

We adopt the name of the potential improvement scheduler introduced in [3], since [15] for the 2-state channel and the previous section for the 3-state channel show that the index value is the ratio of the one-slot holding cost saving and the (weighted) potential improvement of the departure probability. This can be seen also as a way of optimally resolving the trade-off between opportunistic scheduling and short-jobs prioritization, but we note that yet another dimension (the Markovian evolution) comes into play and shows that it may be sometimes better to neither be opportunistic nor give priority to (myopically) shorter job. We can summarize the main features of this rule by saying that the priority is given to users which cannot improve their actual condition by much.

The PI* scheduler and both its approximations $\mathrm{PI}^{\mathcal{AG}}$ and $\mathrm{PI}^{\mathrm{SS}}$ reduce to a scheduler that is optimal if $N_k = 1$ for all $k$ and there is a single server ($C = 1$) under arbitrary arrivals [10]. Also, they belong to the family of the best-condition schedulers, which give always priority to users currently in their best condition over users which are not, and which have important stability properties in Markovian setting as shown in [16].

THEOREM 9. *In the single server case $C = 1$, the PI* scheduler and both its approximations $PI^{\mathcal{AG}}$ and $PI^{SS}$ are maximally stable under arbitrary arrivals.*

We believe that maximal stability is true even in the multi-server case. In fact, it is easy to argue that the stability region is upperbounded even in the case of generally distributed job sizes as follows.

THEOREM 10. *If $\varrho := \sum_{k\in\mathcal{K}} \varrho_k > C$, then there is no scheduler that stabilizes the system.*

We are, unfortunately, unable to conclude anything with respect to (asymptotic) optimality of the proposed schedulers in systems with arrivals. In the next section we evaluate the performance of PI* and compare it to existing schedulers proposed for this problem by previous literature.

## 6. EXPERIMENTAL STUDY

In this section we investigate the behavior of the PI* scheduler and its approximations that we have proposed. In order to be able to evaluate the performance of these policies, we present several scenarios, in which we compare them with the schedulers proposed in previous literature. These policies are all priority-based, in the sense that the users served are the ones with highest index values. We however note that these alternative schedulers are based on indices that are *not* Whittle indices, i.e., they have not been shown optimal in the single-user subproblem.

For the sake of completeness we give their definitions, especially because we have modified them to incorporate the waiting costs (originally equal to 1 for RB, PB and SB):

- the $\boldsymbol{c\mu}$ rule, i.e. $\nu^{c\mu}_{k,n} = c_k\mu_{k,n}$;
- the **Relatively Best** rule, i.e. $\nu^{\mathrm{RB}}_{k,n} = \frac{c_k\mu_{k,n}}{\sum_{m=1}^{N_k} q^{\mathrm{SS}}_{k,m}\mu_{k,m}}$;
- the **Proportionally Best** rule, i.e. $\nu^{\mathrm{PB}}_{k,n} = \frac{c_k\mu_{k,n}}{\mu_{k,N_k}}$;
- the **Score Based** rule, i.e. $\nu^{\mathrm{SB}}_{k,n} = c_k\sum_{m=1}^{n} q^{\mathrm{SS}}_{k,m}$.

We restrict our attention to the case with at most one user served during each slot of time, i.e. $C = 1$ (if more than one user has the highest index value, we break the ties randomly), and we consider only 2 classes of users, in order to be able to easily point out the differences in the performance of the policies generated by the above scheduling rules.

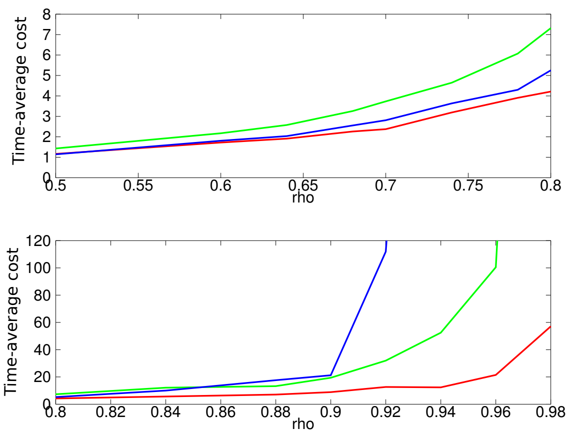
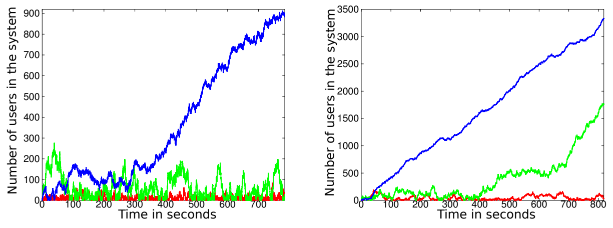


Figure 1: Scenario 1 - Time-average waiting cost of PI*,SB,PB (red), RB (green), $c\mu$ (blue) as a function of varying $\varrho$, computed from simulation over 330 sec.



(a) $\varrho = 0.94$, over 785 sec. (b) $\varrho = 0.98$, over 820 sec.

Figure 2: Scenario 1 - Evolution of the number of users in the system during simulation of PI*,SB,PB (red), RB (green), $c\mu$ (blue).

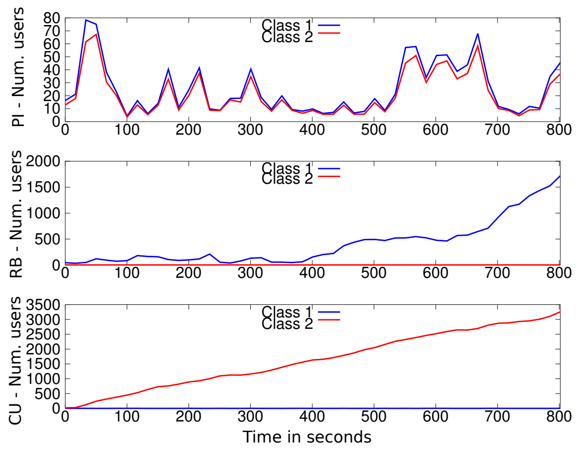


Figure 3: Scenario 1 - Number of users of class 1 (blue) and class 2 (red) for $\varrho = 0.98$, the values are averaged over intervals of 10000 slots (16.7 seconds).

Each class $k$ is characterized by a time independent value $\lambda_k \in [0,1]$, representing the probability that during a slot a new user belonging to class $k$ enters the system. The restriction to Bernoulli arrivals is justified by the shortness of the slot considered, which is $\epsilon = 1.67$msec.

In order to simulate scenarios as realistic as possible we consider transmission rates $s_{k,n}$ employed in LTE networks, see Table 2, which is adapted from [20]. Each class will be identified by one of the following three types of files to be downloaded, typical in a wireless data network:

- HTML web page (or e-mail) with expected job size $\mathbb{E}_0[b_k] = 0.5$Mb (64kB)
- PDF document (or image) with expected job size $\mathbb{E}_0[b_k] = 5$Mb (640kB)
- MP3 audio (or short video) with expected job size $\mathbb{E}_0[b_k] = 50$Mb (6.25MB)

For every class of users we select some channel conditions among the ones defined in Table 2, in this way, we determine the departure probabilities using the formula (1).

Moreover in every simulation we vary the value of $\varrho$ between 0.5 and 1, but for simplicity it is always maintained that $\varrho_1 = \varrho_2$. In this way it is possible to determine the rate of arrivals of the specific class, which is given by the formula $\lambda_k = \varrho_k \mu_{k,N_k}$ for $k = 1, 2$. The channel condition possessed by a user at the moment of his arrival is supposed to be determined by an equidistributed variable among her states, i.e., $q_{k,n} = 1/N_k$.

It is interesting to point out that the condition $c_k = c$ for all $k$ implies that SB and PB are best-condition schedulers. Such a property is guaranteed for the $c\mu$ and $RB$ rule under the condition that respectively the values $c_k\mu_{k,N_k}$ and $\nu^{\text{RB}}_{k,N_k}$ are the same for each user $k$. On the other hand PI* rule generates a BR policy unconditionally. The property of being a best-condition policy is important since it identifies policies that are maximally stable. The parameters for all the scenarios are summarized in Table 3.

## 6.1 Scenario 1

In this scenario the users are divided into two different classes, each user requires a job of expected size 0.5Mb and costs $c_1 = c_2 = 1$ for every slot of waiting. Therefore our objective is to minimize the time-average number of users (uncompleted flows) in the system. The channel condition transition matrix for the two classes is a randomly generated matrix, see Table 3. We suppose that the first class of users has the opportunity to be always served with a better transmission rate than the second class, indeed it can be seen in Table 2 and Table 3 that $s_{1,1} = 53.76$Mb/sec while $s_{2,3} = 33.6$Mb/sec.

It can be checked that in this scenario the rules PI*, SB and PB generate the same policy. Figure 1 shows the time-average waiting cost accrued by employing the different policies for varying $\varrho$. It appears that the behavior of all the policies until $\varrho \leq 0.84$ is quite similar, even if the $c\mu$ and the PI* rules seem to slightly outperform the RB rule. The $c\mu$ rule seems to become unstable between $\varrho = 0.92$ and $\varrho = 0.94$. Indeed in Figure 2 it can be seen that the average increase of users in the system per slot is about 1.2 users per second for $\varrho = 0.94$ (note that the average number of arrivals per second is 75.8 for class 1 and 31.5 for class 2). For such a value of $\varrho$ the other rules are still stable, it appears that RB and PI* rules cost on the average respectively about 60

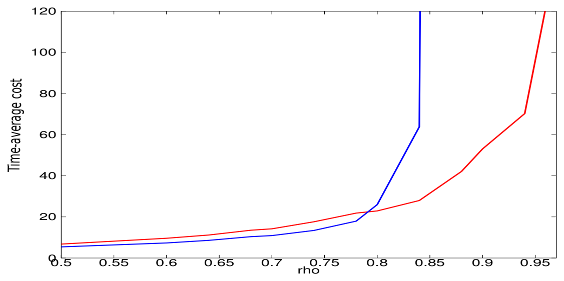


**Figure 4: Scenario 2 - Time-average waiting cost of PI* (red), $c\mu$,SB,PB,RB (blue) as a function of varying $\varrho$, computed from simulation over 330 sec.**

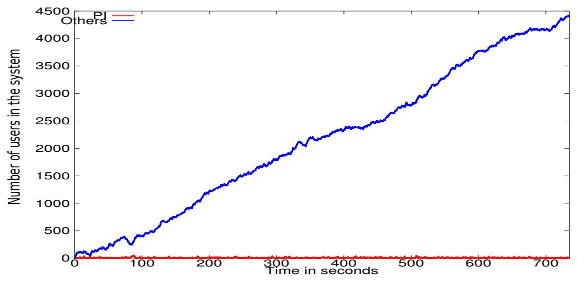


**Figure 5: Scenario 2 - Evolution of the number of users in the system with $\varrho = 0.88$ during simulation of PI* (red), $c\mu$,SB,PB,RB (blue).**

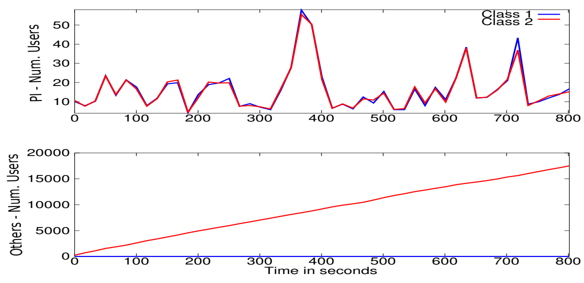


**Figure 6: Scenario 2 - Number of users of class 1 (blue) and class 2 (red) for $\varrho = 0.98$, the values are averaged over intervals of 10000 slots of time (16.7 seconds).**

and 10 per slot. Raising $\varrho$ up to 0.98 (i.e., increasing the average number of arrivals per second to 79 for class 1 and 32.9 for class 2), also the RB policy becomes unstable, and the average increase in the number of users due to the employment of this policy is about 2.1 users per second. It is still better than the average increase caused by the $c\mu$ rule which is of about 4.2 users per second. However these policies are strongly outperformed by the PI* rule, indeed with $\varrho = 0.98$ this rule is still stable and time-average cost is less than 60 per slot. In Figure 3 we display the evolution of the number of users in the system per class. It can be observed that the policies considered behave in a completely different way. In fact, the PI* rule seems to be quite fair between the two classes, the $c\mu$ rule gives priority to the first class users (so that class-2 users accumulate) while on the contrary the RB favors the second one (so that class-1 users accumulate).

## 6.2 Scenario 2

In this scenario we would like to observe the behavior of the different rules when they have to deal with users that are totally identical, except for their "importance". Indeed the users that characterize this scenario possess the same parameters, reported in Table 3. The two classes differ from each other only in the waiting cost, in particular $c_1 = 10$ and $c_2 = 1$. Thus, classes 1 and 2 may represent business vs individual customers, or contracted vs prepaid customers, or proper vs roamed ones.

It is important to notice that in such a scenario the rules RB, SB, $c\mu$ and PB lead to the same policy, the PI* rule is the only one that differs. It can be checked that the policies different from the PI* rules give absolute priority to the users of the first class, so that a user of the second class can be served only when no users of the first class are in the system. Figure 4 displays the average cost per slot on varying $\varrho$, and it appears that the behavior of all the policies until $\varrho \leq 0.8$ is similar. To be more precise, the PI* seems to behave a bit worse than the others. At the same time it is possible to observe that for greater values of $\varrho$ the situation reverses completely, indeed all the policies generated by rules different from PI* do not succeed in avoiding the accumulation of the users (of second class, see Figure 6). In Figure 5 we show the evolution of the number of users in the system during a simulation of about 12 minutes and $\varrho = 0.88$. It can be observed that the average increase of users in the system due to the employment of the rules different from PI* is about 6 users per second (out of 71 arrivals per second for each class). Meanwhile, the policy generated by the PI* rule leads to a stable system even for $\varrho = 0.98$, indeed the average cost is less than 180 per slot. In Figure 6 we show the evolution of the number of users of the different classes in the system during a simulation of around 14 minutes under $\varrho = 0.98$. We can see that the PI* rule does not care so much about the difference of importance between the two classes and keeps the number of users balanced across classes.

## 6.3 Scenario 3

In this scenario the users belonging to the first class require a job (PDF) of expected size ten times bigger than the second class ones (HTML), however, the users of the first class have a better-quality channel (e.g., they are closer to the base station). Moreover, the interesting thing is that the users requiring a smaller service are almost unable to reach their best channel condition. Particular parameters of the two classes are reported in Table 3.

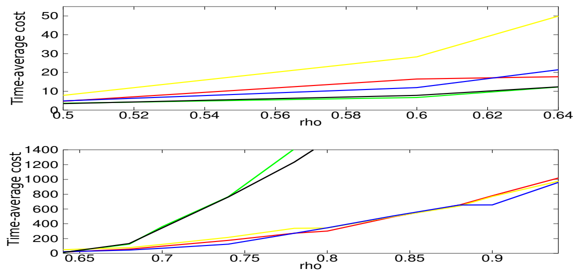
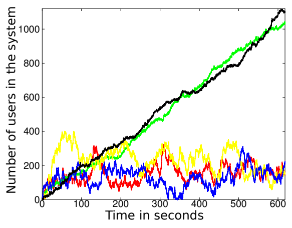
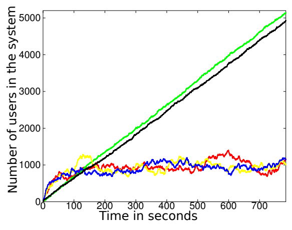


**Figure 7: Scenario 3 - Time-average cost of PI* (red), PB (yellow), SB (blue), RB (green), $c\mu$ (black) as a function of varying $\varrho$, computed from simulation over 330 sec.**



(a) $\varrho = 0.74$, over 618 sec.



(b) $\varrho = 0.94$, over 785 sec.

**Figure 8: Scenario 3 - Evolution of the number of users in the system during simulation of PI* (red), PB (yellow), SB (blue), RB (green), $c\mu$ (black).**

In Figure 7 we report the time-average cost obtained from a 330 seconds simulation. It is possible to see that if the arrivals are quite low, say $\varrho \leq 0.64$, the rules that work better are the $c\mu$ and the RB. These policies become unstable when we increase the probability of new arrivals. Indeed, the evolution of the number of users in the system can be seen in Figure 8, where we display the cases with $\varrho = 0.74$ and $\varrho = 0.94$. The system managed by RB and $c\mu$ rule starts to accumulate users, in particular the number of users in the system increases by about 1.9 users per second already with $\varrho = 0.74$ (there are 6 and 49.7 arrivals per second in each class, respectively). Meantime the other rules maintain stability also for high values of $\varrho$, in Figure 8 it can be seen that they behave quite similarly even for high values of $\varrho$, though they are not equivalent. Moreover, notice the interesting feature that the number of users fluctuates around an equilibrium value of 1000, without growing much nor emptying the system.

## 6.4 Scenario 4

In this scenario the users of the first class require the completion of a very big job $b_2 = 50$Mb compared to the second class of jobs which are one hundred times smaller. The jobs required by the first class users are considered slightly more important than the other, therefore $c_1 = 3$ and $c_2 = 1$. The channel condition of the users belonging to the second class evolves in an almost iid way and their best condition

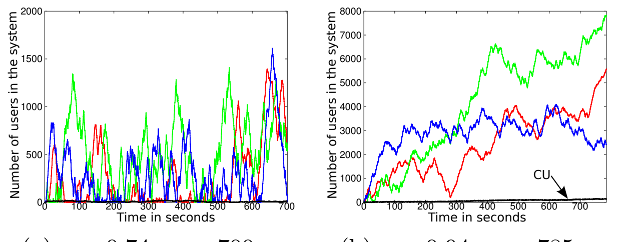

(a) $\varrho = 0.74$, over 700 sec. (b) $\varrho = 0.94$, over 785 sec.

**Figure 9: Scenario 4 - Evolution of the number of user in the system during simulation of $PI*$ (red), $SB$ (blue), $RB, PB$ (green), $c\mu$ (black)**

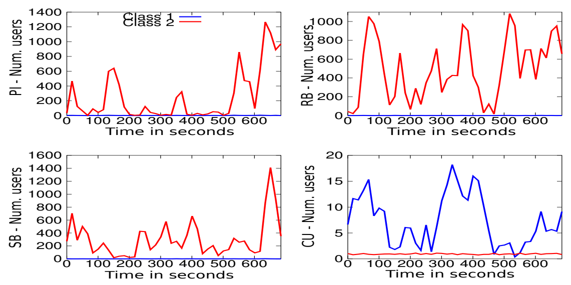


**Figure 10: Scenario 4 - Number of users of class** 1 **(blue) and class** 2 **(red) for** $\varrho = 0.84$**, the values are averaged over intervals of 10000 slots of time (16.7 seconds).**

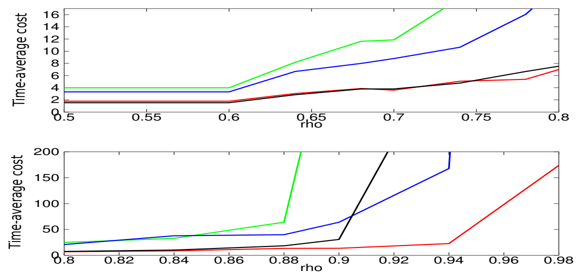


**Figure 11: Scenario 5 - Time-average cost of PI* (red), SB (blue), RB,PB (green), $c\mu$ (black) as a function of varying $\varrho$, computed from simulation over** 330 **sec.**

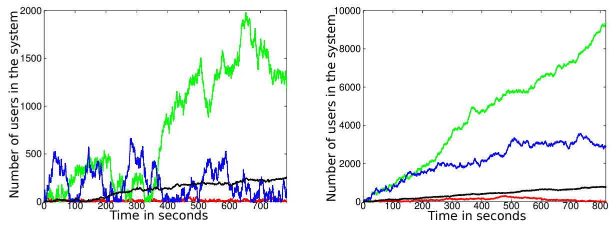

(a) $\varrho = 0.94$, over 785 sec. (b) $\varrho = 0.98$, over 820 sec.

**Figure 12: Scenario 5 - Evolution of the number of users in the system during simulation of PI* (red), SB (blue), RB,PB (green), $c\mu$ (black).**

is almost never reached, as can be seen in Table 3.

With these data, it can be checked that the PB and the RB rules lead to the same policy. This scenario is markedly influenced by the difference in size of the two classes, since for such a reason the departure probabilities of the two classes appear to be very unbalanced. It can be seen that as long as the rate of arrivals (and $\varrho$) is low, all the rules are stable. Nevertheless, there are differences: e.g., by focusing on the case with $\varrho = 0.84$ (with arrivals of 0.6 and 59.7 per second for each respective class), see Figure 9, it is evident that the $c\mu$ rule strongly outperforms the other rules. This fact can be explained by observing that the $c\mu$ rule gives priority to the second class, as it can be seen in Figure 10, and as long as there are not too many arrivals it is also able to serve the users of the other class. So, the $c\mu$ rule queues the small number of large jobs, while the other schedulers queue a big number of small jobs. It is interesting that also for $\varrho = 0.94$ case (with arrivals of 0.76 and 75.8 per second for each respective class) that leads to instability of $c\mu$ rule (see Figure 9), the average increase of users generated by the employment of this rule is only about 0.15 users per second. The other policies appear to be more stable, but it can be checked that both SB and RB are not best-condition (condition 2 of class 1 gets higher priority than condition 3 of class 2), i.e., not maximally stable. However the stability of PI* does not lead to a considerably better performance in practice, and the time-average cost of the policies PI*, RB and SB remains quite high. Such a time-average cost is actually reached (and overcome) by the $c\mu$ rule after about 2-3 hours of service in the system.

## 6.5 Scenario 5

This scenario is easier to analyze than the previous one since the jobs are smaller. Still, we have that the first class of users requires the completion of a job ten times bigger than the second class, and the payment of only $c_1 = 2$. The other parameters are the same for the two classes and are reported in Table 3. It is interesting to point out the structure of the channel condition transition matrix which in this case is diagonally dominant, i.e. the users are most likely to maintain their channel condition from one slot to the following with respect to change it.

In this scenario, like in the previous one, the PB and the RB rules generate the same policy. As can be seen in Figure 11, it happens that for $\varrho \leq 0.9$ the policies generated by rules $c\mu$ and PI* outperform the other policies. Every policy considered is stable until $\varrho = 0.9$, however, the average cost per slot which arise by following the different rules is quite different, indeed it is 10, 15, 60 and 90 respectively for the rules PI*, $c\mu$, SB and RB. It is possible to see in Figure 12 that for $\varrho = 0.94$ (arrivals 7.6 and 75.8 per second) the $c\mu$ rule starts to be unstable, indeed the average increase of users is about 0.3 per second. The other policies are still stable for such $\varrho$ even if RB queues about 1300 users per slot. This cost is reached and overcome by the $c\mu$ rule only after about 1 hour. For $\varrho = 0.98$ (arrivals 8 and 79 per second) also the policy generated by the RB rule starts to have an unstable behavior, see Figure 12, with the employment

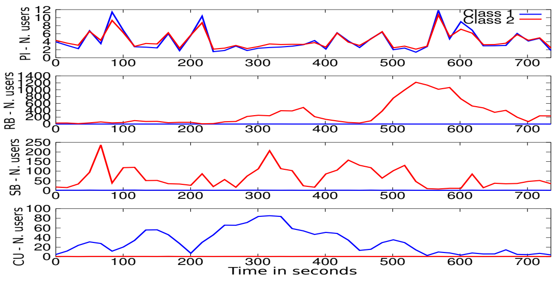


**Figure 13: Scenario 5 - Number of users of class 1 (blue) and class 2 (red) for $\varrho = 0.90$, the values are averaged over intervals of 10000 slots of time (16.7 seconds).**

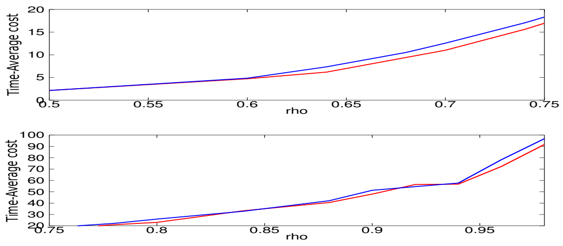


**Figure 14: Scenario 6 - Time-average cost of PI$^{\mathcal{AG}}$ scheduler (red), PI*,SB,RB,PB,$c\mu$ (blue) as a function of varying $\varrho$, computed from simulation over 330 sec.**

of such a policy the number of users in the system increases by almost 12 users per slot, much worse than the 0.95 users per slot that results utilizing the $c\mu$ rule. The policies SB and PI* are still stable for such a value of $\varrho$ even if they queue respectively around 3000 and 200 users per slot. It is really interesting to observe the way in which the different policies treat the two classes. It can be seen in Figure 13 that while the rules SB and PB favor users of the first class and the rule $c\mu$ gives priority to the second class users, the PI* maintains a balance between the two classes.

### 6.6 Scenario 6

In this scenario $\mu$ and $\boldsymbol{Q}$ are chosen so that the problem is not solvable by threshold policies. We create two identical classes, as can be seen in Table 3, so the system is essentially single-class. We have attempted to simulate the behavior of all the policies described above and the one induced by the indices computed through the $\mathcal{AG}$-algorithm, i.e. PI$^{\mathcal{AG}}$ scheduler rule.

This scenario is quite simple and it happens that the PI$^{\text{SS}}$, RB, SB, PB and $c\mu$ rules lead to the same policy. Note that none of them assumes that the non-intuitive order of states (non-optimality of threshold policies) could exist. Only PI$^{\mathcal{AG}}$ captures this phenomenon.

Figure 14 displays the time-average cost that is obtained by averaging a 330 seconds simulation. As expected, PI$^{\mathcal{AG}}$ outperforms the other policies, but the differences between various policies are not significant.

## 7. CONCLUSION

The scheduling problem we are investigating is an elaborate problem and is far from being solved. The introduction of more general and realistic elements, like the Markovian evolution of the channel or the arbitrary number of channel conditions prohibits the possibility to furnish closed, intuitive formulas for the PI* index values. Indeed, mostly due to the Markovian property it is quite a hard job to carry out the probability of moving to a better condition, $q^*$, that we believe plays an important role in the computation of such index values. The problem could be considered almost solved in the 3-state case, where, if the jobs are sufficiently large, the PI$^{\text{SS}}$ scheduler seems to work as an efficient approximation. It should be investigated if such an approximation is still effective for the general case, in particular it could be useful to identify sufficient conditions that guarantee the problem (2) to be solvable by threshold policies.

Numerical simulations suggest that PI* rule works well in a lot of different scenarios. In particular the maximal stability property assures the system to be manageable even under situations of high load. Moreover quite surprisingly even though we have not included fairness optimization in our MDP model, PI* shows to be very fair between classes, and besides it does not depend on $\lambda$ (and $\varrho$).

On the other hand, even only the analysis of the few scenarios reported in the paper establish that in some cases it could be convenient to employ other policies instead of the PI* one. If the system is not loaded much, the $c\mu$ policy, as long as it is stable, performs often better than the other policies. It could be interesting to have knowledge of stability limit of the $c\mu$ policy in order to be able to alternately employ such a policy and the PI*.

The main theoretical limitation of this paper is the assumption of geometric job sizes. However, we believe that it is important first to understand this case, likely to be analytically the simplest, and future research should address the question of (in)sensitivity of our results to the job size distribution. Note also that the existing maximal stability results for best-condition schedulers [4, 16] also rely on the geometric job size assumption.

## 8. REFERENCES

[1] S. Aalto and P. Lassila. Flow-level stability and performance of channel-aware priority-based schedulers. In *Proceeding of NGI 2010 (6th EURO-NF Conference on Next Generation Internet)*, 2010.

[2] S. Aalto, A. Penttinen, P. Lassila, and P. Osti. On the optimal trade-off between SRPT and opportunistic scheduling. In *Proceedings of ACM Sigmetrics*, 2011.

[3] U. Ayesta, M. Erausquin, and P. Jacko. A modeling framework for optimizing the flow-level scheduling with time-varying channels. *Performance Evaluation*, 67:1014–1029, 2010.

[4] U. Ayesta, M. Erausquin, M. Jonckheere, and I. M. Verloop. Scheduling in a random environment: Stability and asymptotic optimality. *IEEE/ACM Transactions on Networking*, 21(1):258–271, 2013.

[5] U. Ayesta and P. Jacko. Method for selecting a transmission channel within a time division multiple access (TDMA) communications system, 2013. EU Patent.

[6] P. Bender, P. Black, M. Grob, R. Padovani,

| **Mod.** | QPSK | | | | | | | | 16QAM | | | | 64QAM | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **MCS** | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| **Rates** | 4.2 | 6.72 | 8.4 | 11.256 | 16.8 | 21.84 | 25.2 | 26.88 | 33.6 | 44.688 | 50.4 | 53.76 | 67.2 | 75.6 | 80.64 |

**Table 2: Transmission rates in Mb/sec associated with LTE modulation and coding schemes (MCS).**

| | MCS | Cost | Channel Condition Transition Matrix | Expected Size |
|---|---|---|---|---|
| #1 | $\{12,13,15\}$, $\{1,8,9\}$ | (1,1) | $\left(\begin{bmatrix} 0.4 & 0.21 & 0.39 \\ 0.48 & 0.5 & 0.02 \\ 0.26 & 0.3 & 0.44 \end{bmatrix}, \begin{bmatrix} 0.34 & 0.35 & 0.31 \\ 0.27 & 0.45 & 0.28 \\ 0.45 & 0.15 & 0.4 \end{bmatrix}\right)$ | (HTML,HTML) |
| #2 | $\{8,12,15\}$, $\{8,12,15\}$ | (10,1) | $\left(\begin{bmatrix} 0.38 & 0.20 & 0.42 \\ 0.43 & 0.19 & 0.38 \\ 0.48 & 0.27 & 0.25 \end{bmatrix}, \begin{bmatrix} 0.38 & 0.20 & 0.42 \\ 0.43 & 0.19 & 0.38 \\ 0.48 & 0.27 & 0.25 \end{bmatrix}\right)$ | (HTML,HTML) |
| #3 | $\{8,12,15\}$, $\{1,9,13\}$ | (1,1) | $\left(\begin{bmatrix} 0.6 & 0.2 & 0.2 \\ 0.3 & 0.5 & 0.2 \\ 0.2 & 0.4 & 0.4 \end{bmatrix}, \begin{bmatrix} 0.5 & 0.499 & 0.001 \\ 0.7 & 0.299 & 0.001 \\ 0.15 & 0.849 & 0.001 \end{bmatrix}\right)$ | (PDF,HTML) |
| #4 | $\{12,13,15\}$, $\{12,13,15\}$ | (3,1) | $\left(\begin{bmatrix} 0.41 & 0.31 & 0.28 \\ 0.16 & 0.5 & 0.34 \\ 0.26 & 0.34 & 0.4 \end{bmatrix}, \begin{bmatrix} 0.65 & 0.34999 & 0.00001 \\ 0.62 & 0.37999 & 0.00001 \\ 0.63 & 0.36999 & 0.00001 \end{bmatrix}\right)$ | (MP3,HTML) |
| #5 | $\{12,13,15\}$, $\{12,13,15\}$ | (2,1) | $\left(\begin{bmatrix} 0.6 & 0.3 & 0.1 \\ 0.25 & 0.5 & 0.25 \\ 0.1 & 0.3 & 0.6 \end{bmatrix}, \begin{bmatrix} 0.6 & 0.3 & 0.1 \\ 0.25 & 0.5 & 0.25 \\ 0.1 & 0.3 & 0.6 \end{bmatrix}\right)$ | (PDF,HTML) |
| #6 | $\{8,9,13\}$, $\{8,9,13\}$ | (1,1) | $\left(\begin{bmatrix} 0.998 & 0.0015 & 0.0005 \\ 0.002 & 0.248 & 0.75 \\ 0.01 & 0.02 & 0.97 \end{bmatrix}, \begin{bmatrix} 0.998 & 0.0015 & 0.0005 \\ 0.002 & 0.248 & 0.75 \\ 0.01 & 0.02 & 0.97 \end{bmatrix}\right)$ | (HTML,HTML) |

**Table 3: Parameters set in the experimental study for (class 1, class 2).**

N. Sindhushayana, and A. Viterbi. CDMA/HDR: a bandwidth-efficient high-speed wireless data service for nomadic users. *IEEE Communications Magazine*, 38(7):70–77, 2000.

[7] T. Bonald. Procédé de sélection de canal de transmission dans un protocole d'accès multiple à répartition dans le temps et système de communication mettant en oeuvre un tel procédé, 2004. EU Patent.

[8] T. Bonald. A score-based opportunistic scheduler for fading radio channels. In *Proceedings of European Wireless*, pages 283–292, 2004.

[9] S. Borst. User-level performance of channel-aware scheduling algorithms in wireless data networks. *IEEE/ACM Transactions on Networking*, 13(3):636–647, 2005.

[10] C. Buyukkoc, P. Varaiya, and J. Walrand. The $c\mu$ rule revisited. *Advances in Applied Probability*, 17(1):237–238, 1985.

[11] E. F. Chaponniere, P. J. Black, J. M. Holtzman, and D. N. C. Tse. Transmitter directed code division multiple access system using path diversity to equitably maximize throughput, 2002. US Patent.

[12] J. C. Gittins. Bandit processes and dynamic allocation indices. *Journal of the Royal Statistical Society, Series B*, 41(2):148–177, 1979.

[13] J. C. Gittins and D. M. Jones. A dynamic allocation index for the sequential design of experiments. In J. Gani, editor, *Progress in Statistics*, pages 241–266. North-Holland, Amsterdam, 1974.

[14] P. Jacko. Restless bandits approach to the job scheduling problem and its extensions. In A. B. Piunovskiy, editor, *Modern Trends in Controlled Stochastic Processes: Theory and Applications*, pages 248–267. Luniver Press, United Kingdom, 2010.

[15] P. Jacko. Value of information in optimal flow-level scheduling of users with Markovian time-varying channels. *Performance Evaluation*, 68(11):1022–1036, 2011.

[16] J. Kim, B. Kim, J. Kim, and Y. H. Bae. Stability of flow-level scheduling with Markovian time-varying channels. *Performance Evaluation*, 70(2):148–159, 2013.

[17] R. Knopp and P. Humblet. Information capacity and power control in single-cell multiuser communications. In *Proceedings of IEEE International Conference on Communications*, pages 331–335, 1995.

[18] H. Kushner and P. Whiting. Convergence of proportional-fair sharing algorithms under general conditions. *IEEE Transactions on Wireless Communications*, 3:1250–1259, 2004.

[19] J. Niño-Mora. Dynamic priority allocation via restless bandit marginal productivity indices. *TOP*, 15(2):161–198, 2007.

[20] S. Sesia, I. Toufik, and M. Baker. *LTE-The UMTS Long Term Evolution: From Theory to Practice.* Wiley, 2011.

[21] P. Whittle. Restless bandits: Activity allocation in a changing world. *A Celebration of Applied Probability, J. Gani (Ed.), Journal of Applied Probability*, 25A:287–298, 1988.