

Decision-Epoch Matters: Unveiling its Impact on the Stability of Scheduling with Randomly Varying Connectivity

N. Soprano-Loto^a, U. Ayesta^{b,c,d}, M. Jonckheere^{a,b}, and I.M. Verloop^b

^aLAAS-CNRS, Université de Toulouse, CNRS, Toulouse, France

^bIRIT, Université de Toulouse, CNRS, Toulouse INP, UT3, Toulouse, France

^cIKERBASQUE, Basque Foundation for Science, 48011 Bilbao, Spain

^dUPV/EHU, Univ. of the Basque Country, 20018 Donostia, Spain

March 28, 2024

Abstract

A classical queuing theory result states that in a parallel-queue single-server model, the maximum stability region does not depend on the scheduling decision epochs, and in particular is the same for preemptive and non-preemptive systems. We consider here the case in which each of the queues may be *connected* to the server or *not*, depending on an exogenous process. In our main result, we show that the maximum stability region now *does* strongly depend on how the decision epochs are defined. We compare the setting where decisions can be made at any moment in time (the unconstrained setting), to two other settings: decisions are taken either *(i)* at moments of a departure (non-preemptive scheduling), or *(ii)* when an exponentially clock rings with rate γ . We characterise the maximum stability region for the two constrained configurations, allowing us to observe a reduction compared to the unconstrained configuration. In the non-preemptive setting, the maximum stability region is drastically reduced compared to the unconstrained setting and we conclude that a non-preemptive scheduler cannot take opportunistically advantage (in terms of stability) of the random varying connectivity. Instead, for the γ decision epochs, we observe that the maximum stability region is monotone in the rate of the decision moments γ , and that one can be arbitrarily close to the maximum stability region in the unconstrained setting if we choose γ large enough. We further show that Serve Longest Connected (SLC) queue is maximum stable in both constrained settings, within the set of policies that select a queue among the connected ones. From a methodological viewpoint, we introduce a novel theoretical tool termed a “test for fluid limits” (TFL) that might be of independent interest. TFL is a simple test that, if satisfied by the fluid limit, allows us to conclude for stability.

Keywords: scheduling, varying connectivity, random modulations, random environments, queuing systems, fluid limits.

1 Introduction

In many real-world applications, scheduling decisions are made at specific moments rather than in a continuous fashion. In telecommunication networks, scheduling decisions for allocating resources and managing bandwidth occur at specific moments to adapt to changing traffic patterns. Other examples include server scheduling where certain maintenance tasks or resource allocation decisions are made periodically, or in smart grids where allocation of resources occur at specific moments based on factors like demand patterns and availability of renewable energy. We can also mention real-time systems where once a task is started, it needs to run without interruption to meet its timing requirements.

It seems intuitively clear that the performance for the end-user will degrade as the interval between the decision epochs increases since the process becomes “less controllable.” To illustrate this, let us consider a canonical multi-class single-server queue where λ_i and μ_i , $i = 1, \dots, K$, denote the arrival and service rates, respectively, and let us compare the system where decisions can be made at any moment in time (preemptive case) versus the system where decisions can only be made upon a departure of a job (non-preemptive case). In both cases, it is known that the μ -priority policy (which chooses to serve the class in the system with highest μ_i) is optimal. Even though the stability condition for both preemptive and non-preemptive systems is the same ($\sum_{i=1}^K \lambda_i/\mu_i < 1$), standard analysis of priority policies ([16, Chapter 5]) shows that the mean queue length of the optimal μ -rule in the preemptive case is indeed larger than in the non-preemptive one.

In this paper, we show that constraining the decision epochs can in fact not only have an impact in terms of steady-state performance, but also on the maximum stability region, that is, on the set of parameters for which there exists a policy which induces a steady-state. We do so by studying a multi-class scheduling problem with time-varying connectivity, a fundamental problem in networking that has been studied in many papers since the pioneering work by Tassiulas and Ephremides [23]. Indeed, time-varying connectivity is inherent in wireless systems and satellite communications [23, 20, 5, 2], where systems face challenges in deciding how to share limited transmitters or channels among users while weather changes make the transmission rates vary significantly. It also arises as a modeling framework in applications where the available capacities fluctuates depending on the traffic conditions. More recently, random environments have also been studied in the context of scheduling in data centers and cloud computing (see for instance [18] or [15]), involving a significant amount of communication overhead such as data transfers, status updates, and other interactions between the client and the cloud-based services, [13]. Those communication overhead being variable, the resulting system can be modelled as queuing systems where each queue can be either connected or disconnected to the server according to a random environment.

We study a continuous-time system with multiple queues that can be either connected or disconnected and there is one server that can be dedicated to at most one queue. However, different from previous work, the server is allowed to change to move to another queue only at certain *specific decision epochs*. We will set out to compare three different settings for these decision epochs. The first setting is when decisions can be made at any moment in time, which we refer to as the unconstrained setting (as studied in [5], referred to as Setting I). The other two are constrained settings where decisions can be taken only at well-defined moments in time: non-preemptive scheduling (referred to as Setting II), or when an exponentially clock rings with rate γ (Setting III). In our main result, we characterise the maximum stability region for the two constrained configurations. The latter are strictly included in the one for the unconstrained setting, see Figure 1. To the best of our knowledge this uncovers a new phenomenon, that is, that the maximum stability region can crucially depend on how decision epochs are defined. We further show that policies that select a queue among the connected ones are not necessarily stable due to the random environments, and prove that the Serve Longest Connected (SLC) queue policy *is* maximum stable for all Settings I, II, and III.

In the non-preemptive case (Setting II), we obtain that the maximum stability region drastically reduces compared to the preemptive case (Setting I). In fact, we will show that it coincides with that of a (classical) multi-class single-server queue where the class- i service rate is reduced to the time-averaged departure rate $\mu_i\pi_i(1)$. The latter allows us to infer that in this setting the scheduler cannot take opportunistically advantage (in terms of stability) of the random varying connectivity.

In Setting III, having decision epochs at an exponential clock with rate γ allows us to model applications where there might be a cost associated in observing the state (and taking the corresponding action). For instance, changing the action too often might be harmful from a performance point of view. Therefore, taking decisions with a controllable rate γ might be a better option than taking

decisions continuously. In this respect, our results reveal that the stability condition of Setting III is strictly smaller than that of Setting I, but converges to Setting I as $\gamma \rightarrow \infty$. Another interesting example is where a reconfiguration delay is incurred every time a decision is taken, which implies that capacity is wasted. The latter means that the frequency of decision epochs that optimizes the performance is non-trivial. Our analysis allows us to find the optimal value for the rate γ such that the maximum stability region is optimized.

Our main proofs rely on fluid techniques. The random environment gives rise to complex averaging phenomena, and as a consequence the fluid limits associated to the Markov process under study are in general very complex and cannot be easily characterized even in low dimensions. On the other hand, the classical quadratic function used as Lyapunov functions for previous models (either for the stochastic version in the simpler case [23] or for the fluid limit [20, 1]) are not directly applicable to our problem at hand. In our analysis, we introduce a novel theoretical tool termed a “test for fluid limits” (TFL), which serves as a simple test that, when satisfied at a fluid scale, allows us to conclude for stability.

Markovian stochastic processes in which actions are taken only at specific decision-epochs, have been analyzed with the so-called *Embedded Markov Chain* approach. Essentially, this boils down to calculating the transition probabilities and the accumulated reward between two decision epochs, which yields a new Markovian process. This has been the traditional approach to study the performance non-preemptive scheduling in queueing models. The novelty of our approach lies in studying how the stability regions of the underlying Markov processes, which can be seen as the most fundamental performance evaluation criteria, are crucially impacted by the definition of these decision epochs.

In summary, our main contributions and findings are:

- We characterize the maximum stability regions and uncover that they crucially depend on how decision epochs are defined.
- We show that the SLC policy is maximum stable in all three settings.
- Policies that select a queue among the connected ones are not necessarily maximally stable.
- In the non-preemptive setting, we prove that the scheduler cannot take opportunistically advantage (in terms of stability) of the random varying connectivity.
- In Setting III, our analysis allows us to find the optimal frequency of decision epochs in order to maximize the stability region.
- We develop a simple test, TFL, that, if satisfied by the fluid limit, ensures stability. This TFL was used for the three different decision epochs settings.

The rest of the paper is organized as follows. In Section 2, we recall seminal results from the state of the art for the multi-class scheduling problem with randomly varying connectivity of the queues. The precise model is described in Section 3. The maximum stability regions for the three different settings of decision epochs are stated in Section 4, as well as the different corollaries and insights that follow from them. In Section 5 we develop a methodological contribution that proves

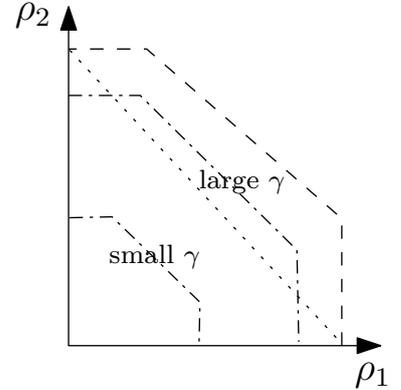


Figure 1: Stability regions with $K = 2$ for Settings I (dashed line), II (dotted line) and III (dash-dotted line, for relatively *small* and *large* γ).

that a simple TFL criterion allows to conclude for stability. Section 6 verifies that this TFL criterion holds for our models allowing to prove our characterization of the main stability regions. Finally, numerical experiments and illustrations are reported in Section 7.

2 Related work

As mentioned in the introduction, models with randomly varying connectivity have been studied in the context of wireless and satellite communications, see [23, 20, 6] just to cite a few from a very large body of literature. Other application domains of these models are in data center and cloud computing, see for instance [18] or [15].

In queueing theory, there is a large body of literature on queues in a random environment. Here, the parameters of the (modulated) queueing model (including arrival rates and service rates) change over time as a function of an exogenous stochastic process. The main focus has been on analyzing the steady-state properties, including stability and queue-length distributions, see for example [3] and [11]. Our model can be seen as a particular instance of a modulated queue, where the service rate of queue i fluctuates between 0 and μ_i .

For the parallel-queue single-server model, a seminal paper is [23] who considered a slotted system with Markovian assumptions and establishes necessary and sufficient conditions for stability. It further showed that SLC is maximum stable and, in the case of symmetric queues, minimizes delay. The proof technique uses a quadratic Lyapunov function. In [5], a Loyne’s construction was employed for an equivalent continuous-time system, wherein the scheduler possesses knowledge of either the workloads or queue-lengths. We also refer to [22] where it is shown that, under the same stability conditions of [23], there exists a so-called Static Service Split rule that is stable. Similar stability conditions were derived in [20] and [1] for more general models (the environment can depend both on classes and the server), and these results were extended to various scheduling types, including variants of the SLC policy and versions of the Max Weight policy (see the book [21] for a pedagogical review on these models). See also [9] for the stability analysis of a model with switching delays.

It is important to highlight that the fluid limit characterizations derived in [20] and [1], may not be easily generalized to our context due to the significantly more complex dynamics stemming from the presence of decision epochs during which multiple events can unfold. We further note that in all works cited above, the stability results are for models where the decisions can be made at any moment in time and the state of the environment is fully known to the controller. However, to the best of our knowledge, no stability results have been obtained when decision epochs are constrained (as in our Setting II and Setting III).

One of our main results states that SLC is maximum stable in Settings II and III. Interestingly, SLC is not always maximum stable, as shown for example in [12] where there are constraints in the set of queues the scheduler can serve. We also refer to [17], where it is shown that SLC can yield arbitrarily low throughput in a wireless setting with signal interference.

Finally, let us mention that there are also several works that deal with an unobservable random environment, that is, when the decision maker cannot observe the state of the environment and it can take its decision only based on the state of the queues. In [14] it is shown that as the number of queues grow large and the environment changes state relatively fast, an index policy is asymptotically optimal. Another interesting work is [8], where the authors study a single-server multi-class queueing network in heavy traffic with randomly varying rates. In the main result, it is shown that an “averaged” version of the classical $c\mu$ -rule is asymptotically optimal.

3 Model description

3.1 Multi-class scheduling in a random environment

The system consists of K parallel queues and a single server. We call $[K] = \{1, \dots, K\}$ the set of queues. Each queue $i \in [K]$ has associated an arrival rate $\lambda_i > 0$, and a service rate $\mu_i > 0$. For $i \in [K]$, we define the variable $\rho_i = \lambda_i/\mu_i$, usually referred to as the load of the i -th queue. In addition, each queue $i \in [K]$ can be either connected or disconnected. Whether or not a queue is connected does not affect the arrivals, but when it is disconnected it cannot receive service. Formally, the environmental state-space associated to each queue $i \in [K]$ is $\{0, 1\}$, the 0 (resp. the 1) representing that the i -th queue is disconnected (resp. connected). The environment of each queue has its own law of evolution: for $i \in [K]$, the environment of queue i passes from 0 to 1 (resp. from 1 to 0) at rate λ'_i (resp. at rate μ'_i). We denote by π_i the invariant distribution of the environment of queue i , which equals

$$\pi_i(0) = \frac{\mu'_i}{\lambda'_i + \mu'_i} \quad \text{and} \quad \pi_i(1) = \frac{\lambda'_i}{\lambda'_i + \mu'_i}.$$

At each moment in time, the server can be dedicated to at most one queue, the service speed being μ_i if queue i is connected and is being served. Decision of which queue to serve can be made only on well-defined *decision epochs*. This decision can not be changed until a next decision epoch. There is also an independent of everything else sequence of times, with inter-arrival frequency given by a parameter $\gamma > 0$, that can determine the decision epochs. Decision epochs will be explained in detail in Section 3.2. For simplicity of exposition, we will assume that all times between events —i.e. arrivals, services, changes in environments and the γ -intensity events just mentioned— are exponential and mutually independent.

The state space of our process is

$$\mathcal{X} = \mathbb{N}_0^K \times \{0, 1\}^K \times \{1, \dots, K, \emptyset\}. \quad (3.1)$$

We are adopting the conventions $\mathbb{N} = \{1, 2, \dots\}$ and $\mathbb{N}_0 = \{0, 1, 2, \dots\}$ respectively for the sets of natural numbers and non-negative integers. The three entries in (3.1) respectively represent the number of waiting tasks per queue, the states of the environments, and the queue that is receiving service. We will sometimes refer to the third entry as the state of the server. If the state of the server is \emptyset , this means that none of the queues receives any service. A state will be denoted as $x = (q, e, c)$, and for a given policy and given decision epochs, the process with initial condition x will be denoted as

$$(X^x(t))_{t \geq 0} = ((Q^x(t), E^x(t), C^x(t))_{t \geq 0}.$$

We highlight that the distribution of the process depends on the policy applied and the decision epochs considered. However, we chose not to include this information in the notation in order to avoid making it too overloaded, trusting that the context is sufficient to avoid any possible confusion. At decision epochs t , the policy is assumed to make decisions based only on the pair $(Q^x(t), E^x(t))$, namely from the present time observation of the queue sizes and the states of the environment, in particular implying that the process is Markov.

Remark 1 (Markovian assumptions). *We opted for Markovian assumptions on the dynamics because of an escalation in technical difficulty without a significant gain in insights. In particular, dealing with more general distributions would give rise to non-explicit and intricate expressions of the key parameters, while their Markovian counterparts are simple.*

3.2 Decision epochs

We describe now the three different settings for decision epochs:

Setting I. Decision epochs are at any moment in time. Stability under this setting has been studied originally in discrete time in [23], while continuous-time versions and generalisations have been considered in [4, 20] (see Section 2 on related).

Setting II. This is the non-preemptive situation in which a decision cannot be taken in the middle of a task. In other words, decision epochs are any moment after a departure and until a new task enters into service.

Setting III. Decision epochs are when an exogenous exponential clock of intensity $\gamma > 0$ rings.

In the context of queues with random connectivity, to the best of our knowledge, Setting I has been the only case studied so far. We can mention [23, 20, 5, 2] and other references in Sections 1 and 2. Setting II is motivated by applications in which a decision cannot be taken in the middle of a task. This occurs for instance in real-time systems where once a task is started, it needs to run without interruption to meet its timing requirements. Finally, Setting III aims at modeling situations in which there might be a cost associated to observing the state or to taking a decision. Here, the administrator might need to define the scheduling epochs at some intervals (defined by clock intensity γ) in order to strike a good balance between performance and cost (both being increasing as a function of γ).

3.3 Scheduling policies and maximum stability regions

We say that a policy stabilizes the system if its associated Markov process is positive recurrent or, equivalently, if it has a unique invariant distribution. For $J \in \{I, II, III\}$, let MSR_J be the maximum stability region associated to Setting J, which is defined as follows:

$$MSR_J = \{(\lambda, \mu, \lambda', \mu') : \text{there exists a policy that stabilizes the system under Setting } J\}.$$

We are using the compact notation $\lambda = (\lambda_1, \dots, \lambda_K)$, $\mu = (\mu_1, \dots, \mu_K)$, and so on. A policy is called *maximum stable* if it can stabilize the system for any set of parameters $(\lambda, \mu, \lambda', \mu') \in MSR_J$.

Characterizing the maximum stability region for Setting III, MSR_{III} , seems out of reach. We refer to Section 4.6 for a full discussion on this. In order to obtain a closed-form expression for the maximum stability region, we will restrict the search to the following class of policies, denoted by \mathcal{P} . For insights as to why we restrict to this set, we refer to the text right above Section 4.1.

Definition 3.1 (class \mathcal{P} of policies). *We say that a policy is inside the class \mathcal{P} if at decision epochs it can only choose to serve a queue that is connected. In addition, it will choose a connected queue with waiting tasks, whenever possible. The policies in this family will be called \mathcal{P} -policies.*

Note that when no queue is connected at a decision epoch, no queue is served and hence the state of the server is \emptyset , until the next decision epoch. An important policy inside the class \mathcal{P} is the Serve the Longest Connected (SLC) policy, as first defined in [23]. SLC is defined as the policy that *at each decision epoch* chooses to dedicate the server to the queue with the highest number of waiting tasks among the ones that are connected, with an arbitrary tie-breaking rule.

The maximum stability region under Setting $J \in \{I, II, III\}$ when restricted to policies in the family \mathcal{P} is defined as:

$$MSR_J^{\mathcal{P}} = \{(\lambda, \mu, \lambda', \mu') : \text{there exists a } \mathcal{P}\text{-policy that stabilizes the system under Setting } J\}.$$

We will write $MSR_{III}(\gamma)$ to show the dependence on γ in the case of Setting III.

4 Main stability results

In this section, we state our main result, which gives an explicit characterization of the maximum stability regions in all three settings.

Theorem 4.1. *For $L \subseteq [K]$, let*

$$\pi_L^0 = \prod_{i \in L} \pi_i(0)$$

represent the probability that all the environments of queues in L are disconnected. We then have that

1. MSR_{I} and $\text{MSR}_{\text{I}}^{\mathcal{P}}$ are characterized by

$$\sum_{i \in L} \rho_i < 1 - \pi_L^0 \quad \forall L \subseteq [K], L \neq \emptyset. \quad (4.1)$$

2. MSR_{II} and $\text{MSR}_{\text{II}}^{\mathcal{P}}$ are characterized by

$$\sum_{i=1}^K \frac{\rho_i}{\pi_i(1)} < 1. \quad (4.2)$$

3. Finally, $\text{MSR}_{\text{III}}^{\mathcal{P}}(\gamma)$ is characterized by

$$\sum_{i \in L} \frac{\rho_i}{\theta_i(\gamma)} < 1 - \pi_L^0 \quad \forall L \subseteq [K], L \neq \emptyset, \quad (4.3)$$

where

$$\theta_i(\gamma) = \frac{\gamma + \lambda'_i}{\gamma + \lambda'_i + \mu'_i}.$$

In all the three settings, the corresponding maximal stability regions are attained by the SLC policy.

For Setting I, both the maximum stability condition and the stability of SLC were first obtained in [23] (see also [5]), while the results for Settings II and III are new. The proof of Theorem 4.1 can be found in Section 6. We note that the proof (for all three settings) is based on a novel methodological tool termed TFL, which serves as a simple test to conclude for stability, see Section 5.

Intuitively, the maximum stability regions under the three settings can be explained as follows. Consider a subset of classes L . The right hand side (RHS) represents the maximum fraction of time the server can be usefully dedicated to this set L in a saturated regime and when queues in L are given priority over the other queues. The left hand side (LHS) of the conditions represents the *effective load* of this subset L , which is defined as $\sum_{i \in L} \lambda_i / (\xi_i^{\text{J}} \mu_i)$, $\text{J} \in \{\text{I}, \text{II}, \text{III}\}$. Here, ξ_i^{J} represents the proportion of time the i -th queue is connected within the time the server is usefully dedicated to it. In this way, $\xi_i^{\text{J}} \mu_i$ represents the effective departure rate. The proportion ξ_i^{J} appears because, due to the restrictions on decision making, there is a proportion of time in which the server is stuck in a disconnected queue, even though other queues in L are connected. This phenomena does not occur in Setting I, reason why $\xi_i^{\text{I}} = 1$. The three conditions now follow:

- Setting I: The RHS is $1 - \pi_L^0$, since it is only useful to dedicate the server to the set L when at least one queue in L is connected. As already mentioned, since decision epochs are any moment in time, the server only dedicates service to a queue when it is connected, so that the effective departure rate of a class is μ_i , which explains the LHS.

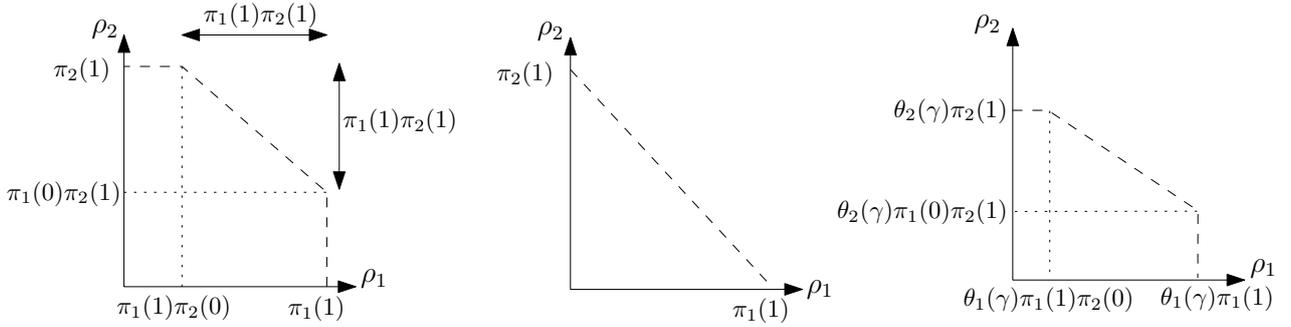


Figure 2: For $K = 2$, we depict MSR_I (left), MSR_{II} (center) and MSR_{III} (right).

- Setting II: Since the service is non-preemptive, we are sure that at a decision epoch the queue under service is connected. We can therefore stay serving a queue in the set L , since for sure there was one queue in L connected at the decision epoch. The fraction of time the server can be dedicated to a subset L is therefore simply 1, that is, the RHS. We briefly explain how the quantity $\xi_i^{II} = \pi_i(1)$ can be obtained. For a fixed queue i , the process encoding the state of its environment (connected or disconnected) only when the server is dedicated to this queue, turns out to be a Markov process whose invariant distribution is precisely π_i . Then the ergodic theorem implies that the desired proportion ξ_i^{II} is $\pi_i(1)$. A more detailed explanation is provided in Section 6.1.

The reason why the maximum stability condition does not depend on subsets of queues is that, if the stability condition is satisfied for the total queue subset $[K]$, then it is automatically fulfilled for any other subset L , since the RHS does not depend on the subset of queues L considered.

- Setting III: At a decision moment, the server can be dedicated to a queue in L only if one of the queues is connected (recall that we are restricting to \mathcal{P} -policies). The fraction of time in which this occurs is $1 - \pi_L^0$ because the law governing the decision epochs (the exponential clocks of rate γ) is independent of the law governing the environments. This explains the RHS. The reason why $\xi_i^{III} = \theta_i(\gamma)$ is similar to that in Setting II, in this case the Markov process at issue having an invariant distribution that depends on γ . A detailed explanation can be found in Section 6.2.

Our result shows that the maximum stability region strongly depends on the chosen setting of the decision epochs. In particular, the MSR_{II} and $\text{MSR}_{III}^{\mathcal{P}}(\gamma)$ are strict subsets of MSR_I as expected, since the decision epochs are more restricted. We refer to Figure 1 for a plot of the different maximum stability regions for the case $K = 2$.

There is a small caveat concerning Setting III. In fact, the maximum stability region is obtained when restricting to policies in \mathcal{P} only. In Section 4.6, we will prove that policies outside this class can outperform the \mathcal{P} -policies in terms of stability. That is, from the stability point of view, it might be better to serve a queue that is currently disconnected, even though there is a connected queue with work waiting. This counter-intuitive property results from the fact that after a decision is made, disconnected queues can become connected (before the next decision epoch), and hence the occasional strategic choice of serving a currently disconnected queue can therefore optimize for higher departure rates in the future. Hence, the actions taken under a maximum stable policy should be a function of whether or not queues are connected which complicates significantly the analysis.

Due to this complexity, we leave the search for such a maximum stable policy, as well as finding the maximum stability region $\text{MSR}_{\text{III}}(\gamma)$, for future research and focus here on $\text{MSR}_{\text{III}}^{\mathcal{P}}(\gamma)$ instead.

We note that restricting to the set of policies \mathcal{P} is relevant. Firstly, this allows us to obtain interesting results such as the characterization in closed-form of the maximum stability region and in showing that SLC is maximum stable. Secondly, our results show that \mathcal{P} is somehow a “complete” class of policies, since by taking γ large enough its stability region can be arbitrarily close to the maximum stability region in the unconstrained setting (Setting I).

4.1 Are \mathcal{P} -policies always maximum stable?

In the classical multi-class scheduling problem, that is, when all the queues are always connected, and decisions can be taken at any moment in time, it is well known that any *work-conserving policy* is stable in the maximum stability region. In fact, this is valid for both preemptive and non-preemptive systems. A natural question is whether this remains valid when adding either (a) varying connectivity or (b) decision restrictions defined by γ . We note that in the presence of (a) and/or (b), the set of policies \mathcal{P} is the equivalent of work-conserving policies in the classical queue. Thus, in this section we aim at answering the following question, **Q1: Is it true that any \mathcal{P} -policy is maximum stable in the presence of (a) or (b)?** We note that Setting I and Setting II would correspond to adding (a) to a preemptive and non-preemptive system, respectively, while Setting III would correspond to having both (a) and (b). When instead we add only (b), we retrieve a classical multi-class scheduling problem where decisions can be taken only at γ -epochs, which we refer to as Setting 0 (with slight abuse of notation). In the proposition below, we show that the answer to **Q1** is negative for all cases. The underlying reason being that, unlike in the classical case, by adding (a) and/or (b) to the system, for stability it is important to keep queues balanced. That is, it is risky to have queues with a small number of tasks, because in the case of (a) this leads to states where all connected queues are empty and we have to idle unnecessarily, or in the case of (b) deciding at a γ -moment to serve a connected queue that has very little tasks brings the risk that the queue empties before the next decision moment and as such the server is again unnecessarily idle.

Proposition 4.2. *Given a decision epoch Setting J , with $J = 0, \text{I}, \text{II}$ or III , and given a priority ordering of the queues. We consider the priority policy that at each decision moment serves the queue with highest priority among all connected queues with waiting tasks. Then, there exist parameters inside the maximum stability region $\text{MSR}_J^{\mathcal{P}}$ for which this priority policy is not stable.*

This provides the negative answer to **Q1** because the priority policy defined in Proposition 4.2 is inside the class \mathcal{P} , but it is not maximum stable. Below we give the proof for $J = \text{I}$, the other settings follow in a similar fashion.

Proof. Assume we are in Setting I and w.l.o.g. assume we have the priority ordering $1 \succ 2 \succ \dots \succ K$. We consider π_i , $i = 1, \dots, K$, fixed and take $\rho_1 \approx 0$. Since $\rho_1 < 1 - \pi_1(0)$, the dynamic of the sole queue 1 is stable, so that we can define the asymptotic proportion of time $\varepsilon_1 > 0$ that this queue is non-empty and connected. Since queue 1 has priority, queue 2 can only be served when queue 1 is not served, which happens with a fraction of time $1 - \varepsilon_1$. Queue 2 can use this time only if it is connected, hence, it is stable if and only if

$$\rho_2 < (1 - \pi_2(0))(1 - \varepsilon_1). \quad (4.4)$$

Observe that ε_1 has been constructed independently of ρ_2 , so at this point we can chose ρ_2 in such a way that condition (4.1) is fulfilled for $L = \{2\}$ and $L = \{1, 2\}$, while (4.4) is not. Now setting also $\rho_3 = \dots = \rho_K \approx 0$, we can guarantee that all conditions in (4.1) are fulfilled. Hence, even though the parameters are inside the maximum stability region MSR_{I} , the priority policy is not stable. \square

4.2 Non-preemptive versus preemptive scheduling

In the non-preemptive case (Setting II), Theorem 4.1 shows that the maximum stability region drastically reduces compared to the preemptive case (Setting I). As explained previously, in Setting II, the stability region coincides with that of a (classical) multi-class single-server queue in which the class i service rate is $\mu_i\pi_i(1)$.

The reduction in stability due to the non-preemption assumption can be calculated as follows: From Theorem 4.1 it follows that the maximum stability region is a convex-hull whose volume can be computed. In order to assess the degradation of the stability region we compare VOL_I and VOL_{II} , where VOL_I (VOL_{II}) denotes the volume of the space enclosed by points (ρ_1, \dots, ρ_K) that satisfy MSR_I (MSR_{II}). In the case $K = 2$, see Figure 2, we can calculate these volumes: $\text{VOL}_I = \pi_1(1)\pi_2(1) - \frac{(\pi_1(1)\pi_2(1))^2}{2}$ and $\text{VOL}_{II} = \frac{\pi_1(1)\pi_2(1)}{2}$, which gives as reduction in stability

$$\text{Red}_{II} := \frac{\text{VOL}_I - \text{VOL}_{II}}{\text{VOL}_I} = \frac{1 - \pi_1(1)\pi_2(1)}{2 - \pi_1(1)\pi_2(1)},$$

for $\pi_1(1), \pi_2(1) > 0$, due to the non-preemptive assumption. This shows that the reduction does not depend on the arrival rates or departure rates. It does however depend on the stationary distribution of the connectivity of the queues. We see that the reduction is close to zero when the queues are almost always connected, while it is close to $1/2$ when the queues are almost always disconnected. To illustrate the latter, consider Figure 2 and observe that MSR_I converges to a square (whose volume converges to zero), and the border of MSR_{II} is the diagonal of this square. Hence, in the limit we obtain as reduction $1/2$.

For arbitrary K , MSR_I and MSR_{II} are polytopes defined by the convex hull of the vertices obtained from linear inequalities. We note that there is no analytical expression to calculate the volume of a convex polytope in general. However, MSR_{II} has a special structure, a simplex, for which the general formula for the volume is $\text{VOL}_{II} = \frac{1}{K!} \prod_{i=1}^K \mu_i\pi_i(1)$. Unfortunately, since we do not have any expression for VOL_I , Red_{II} can only be estimated by numerical means.

4.3 Different time scale regimes

In this section, we study the impact that the speed of the environment has over the stability regions. For that purpose, we use a speed factor $\alpha > 0$ which acts over the system by multiplying the rates λ'_i and μ'_i for each i . The first observation is that MSR_I and MSR_{II} remains unaffected under the scaling given by α . In Setting III, where this is not the case anymore, we will study the impact that this scaling has in combination with the parameter γ .

- If $\frac{\alpha}{\gamma} \rightarrow 0$, then from Theorem 4.1 it follows that

$$\lim_{\frac{\alpha}{\gamma} \rightarrow 0} \text{MSR}_{III}^{\mathcal{P}}(\gamma) \rightarrow \text{MSR}_I.$$

This can be explained because, in this limiting regime, decision epochs are so often compared to changes in the environment that we approach Setting I in which decisions can be made at every moment.

- If $\frac{\alpha}{\gamma} \rightarrow \infty$, then from Theorem 4.1 it follows that

$$\lim_{\frac{\alpha}{\gamma} \rightarrow \infty} \text{MSR}_{III}^{\mathcal{P}}(\gamma) \rightarrow \sum_{i \in L} \frac{\rho_i}{\pi_i(1)} < 1 - \pi_L^0 \quad \forall L \subseteq [K], L \neq \emptyset.$$

It can be easily checked that the latter is strictly included in MSR_{II} . To explain this inclusion, observe that, in the limiting regime, the states of connectivity of the queues changes infinitely

many times between two decision epochs, and as such one observes as effective departure rate the steady state $\mu_i \pi_i(1)$. However, contrary to Setting II, the fraction of time that can be dedicated to a set of queues L is given by $1 - \pi_L^0$, since only queues that are connected at a γ moment are allowed to receive service, which explains the strict reduction in stability in the RHS.

4.4 Sufficiently large frequency to guarantee stability

It follows from Theorem 4.1 that $\text{MSR}_{\text{III}}^{\mathcal{P}}(\gamma)$ is monotone increasing on γ , and in particular, $\text{MSR}_{\text{III}}^{\mathcal{P}}(\gamma) \rightarrow \text{MSR}_{\text{I}}$ as $\gamma \rightarrow \infty$, see Section 4.3. In some applications, there might be a cost (for instance energy consumption) associated to observing the state. In such a situation, the system administrator might want to keep the frequency γ , as low as possible, while keeping the system stable. Theorem 4.1 ensures the existence of a minimal value for γ , denoted by γ_0 , such that the system can be made stable for any $\gamma > \gamma_0$.

Corollary 4.3. *For a set of parameters $(\lambda, \mu, \lambda', \mu') \in \text{MSR}_{\text{I}}$, there exists a finite γ_0 such that for $\gamma > \gamma_0$, the SLC policy is stable in Setting III.*

Equivalently to Subsection 4.2, we define $\text{VOL}_{\text{III}}(\gamma)$ as the volume of the space enclosed by points (ρ_1, \dots, ρ_K) that satisfy $\text{MSR}_{\text{III}}^{\mathcal{P}}(\gamma)$, and $\text{Red}_{\text{III}}(\gamma) := \frac{\text{VOL}_{\text{I}} - \text{VOL}_{\text{III}}(\gamma)}{\text{VOL}_{\text{I}}}$ as the reduction of stability for Setting III compared to Setting I. Even though we cannot calculate the volumes MSR_{I} and $\text{MSR}_{\text{III}}^{\mathcal{P}}(\gamma)$ for $K > 2$, it turns out we can instead directly calculate $\text{Red}_{\text{III}}(\gamma)$. To see this, we note that if (x_1, x_2, \dots, x_K) is a vertex of MSR_{I} , then $(\theta_1(\gamma)x_1, \theta_2(\gamma)x_2, \dots, \theta_K(\gamma)x_K)$ is a vertex of $\text{MSR}_{\text{III}}^{\mathcal{P}}(\gamma)$ as well. We refer to Figure 2 for an illustration when $K = 2$. This then allows us to relate their volumes, see the following corollary. For the proof, we refer to the Appendix C.

Corollary 4.4. *For arbitrary K , we have*

$$\text{Red}_{\text{III}}(\gamma) := \frac{\text{VOL}_{\text{I}} - \text{VOL}_{\text{III}}(\gamma)}{\text{VOL}_{\text{I}}} = 1 - \theta_1(\gamma)\theta_2(\gamma)\cdots\theta_K(\gamma). \quad (4.5)$$

Since $\theta_i(\gamma)$ is increasing in γ , it follows that $\text{Red}_{\text{III}}(\gamma)$ decreases as γ increases. We can then consider a similar example to the one that led to Corollary 4.3. Let us assume that there is cost associated to observing the state. A system administrator might then want to determine the rate γ_1 sufficiently large so that the stability reduction is smaller than a certain threshold.

Corollary 4.5. *For a set of parameters $(\lambda, \mu, \lambda', \mu') \in \text{MSR}_{\text{I}}$ and a target stability reduction R , there exists a finite γ_1 such that $\text{Red}_{\text{III}}(\gamma) \leq R$, for all $\gamma \geq \gamma_1$.*

4.5 Optimizing the maximum stability region in the presence of communication overhead

Assume that each decision-making incurs a communication overhead, leading to the server being momentarily diverted from task execution and engaged in a communication process. To make things simpler, let us further assume that each decision-making then incurs an inactivity of fixed duration c before the server resumes. For this model, the maximum stability region can be derived from $\text{MSR}_{\text{III}}^{\mathcal{P}}(\gamma)$ and is given in the following Corollary (see the Appendix C for the proof).

Corollary 4.6. *Consider Setting III. Assume that at each decision epoch the server incurs an inactivity period of fixed duration c . The precise scheduling decision of which queue to serve is decided after this inactivity period. The maximum stability region is then given by*

$$\sum_{i \in L} \frac{\rho_i}{\theta_i(\gamma)} < \frac{1 - \pi_L^0}{1 + \gamma c} \quad \forall L \subseteq [K], L \neq \emptyset.$$

It then follows that for a given set of parameters, there exists a finite γ_c^* such that the maximum stability region is maximized in γ_c^* .

The value of γ_c^* corresponds to the optimal rate of decision epochs that strikes the right balance between a high frequency of taking actions and the overheads induced by them.

4.6 Could SLC be improved upon?

The SLC policy was shown to be maximum stable when decisions can be made at any moment in time (Setting I), as well as when decisions have to be non-preemptive (Setting II), see Theorem 4.1. When decisions happen at γ -moments (Setting III), we proved that SLC is maximum stable when restricting to the set of \mathcal{P} -policies. The question remains whether SLC is also maximum stable in general, that is, stable in the set $\text{MSR}_{\text{III}}(\gamma)$. In this section, we show that the answer to this question is negative and discuss other policies that could potentially be maximum stable.

To show that SCL is not maximum stable in Setting III, we prove that the maximum stability region $\text{MSR}_{\text{III}}(\gamma)$ is strictly larger than $\text{MSR}_{\text{III}}^{\mathcal{P}}(\gamma)$. This is stated in Proposition 4.7 and for its proof we refer to Appendix C. Intuitively, the first inclusion in (4.6) can be seen as follows: we will define a policy outside the family \mathcal{P} which improves SLC in terms of stability. That is, there exist sets of parameters outside $\text{MSR}_{\text{III}}^{\mathcal{P}}(\gamma)$ for which this new policy is stable. The policy is defined as follows: at a decision epoch, its action is identical to SLC if there is at least one queue that is connected; otherwise, when all the queues are disconnected, it dedicates the service to the queue with the highest number of tasks instead of going to the \emptyset state as SLC does. The improvement in terms of stability comes from the fact that the intervals delimited by consecutive γ -marks at which SLC is in state \emptyset represent a positive proportion of time which is seized by the new policy.

Proposition 4.7. *It holds that*

$$\text{MSR}_{\text{III}}^{\mathcal{P}}(\gamma) \subset \text{MSR}_{\text{III}}(\gamma) \subset \text{MSR}_{\text{I}}. \quad (4.6)$$

Here \subset denotes a subset in the strict sense.

The second inclusion in the above proposition states that $\text{MSR}_{\text{III}}(\gamma) \subset \text{MSR}_{\text{I}}$. That is, the timing restriction in Setting III creates a strict difference in terms of stability. Nevertheless, as stated in Corollary 4.3, in Setting III there exist policies (e.g. SLC) that are stable for parameters in MSR_{I} as the decision rate γ is set sufficiently large. A full characterization of $\text{MSR}_{\text{III}}(\gamma)$ is left as future research.

Another question that is left unanswered is which policies *do* stabilize the system for parameters in $\text{MSR}_{\text{III}}(\gamma)$. For now, we do not have an answer to this. We do believe however that *a maximum stable policy will need to occasionally serve a disconnected queue* even though there are connected queues with tasks waiting to be served¹: Although choosing a disconnected queue is disadvantageous in the short term, it might be beneficial in the long run. The latter follows because between two decision epochs the queue might become connected and hence be able to serve tasks. If the disconnected queue had a large backlog while the connected queue had very few tasks waiting, serving the disconnected queue might be the preferred action in order to avoid unbalanced queues (see Section 4.1 for an explanation as to why one would want to avoid unbalanced queues).

We conclude this section by mentioning two type of policies that might be able to provide a maximum stable system in Setting III. The first one is a static policy (that is, whose actions do not depend on the queue lengths) defined by a so-called Static Service Split (SSS) rule as introduced

¹Note that the earlier mentioned “new version” of SLC was only different to SLC when all queues were disconnected. This was sufficient to prove the strict inclusion of $\text{MSR}_{\text{III}}^{\mathcal{P}}$ in MSR_{III} , but it will not be enough for the policy to be maximum stable.

in [22] for a more general environment model (without restrictions on the decision epochs). Under the SSS policy, each possible set of connected queues is associated a probability vector (the static service split) that determines with which probability each of the queues is chosen to be served at a decision epoch when only this set of queues is connected. For Setting I, stability of SSS rules was proved in [22, Proposition 1]. More precisely, it is proved that for each set of parameters in the maximum stability region MSR_I there exists a static service split such that the SSS rule is stable in Setting I. We believe that similar results hold true for $\text{MSR}_{\text{III}}(\gamma)$ in Setting III, but leave this for future research. Another candidate for a stable policy we would like to mention is based on the restless bandit framework as introduced by Whittle in [26]. In this framework, a queue i is seen as an arm whose two-dimensional state consists of the number of waiting tasks, $Q_i(t)$, and whether or not the queue is connected, $E_i(t)$. Each queue is associated a Whittle index as a function of $Q_i(t)$ and $E_i(t)$. The so-called Whittle index policy consists in serving at each decision epoch the queue that has currently the highest Whittle index value. Results in the literature show that such a policy can be very efficient, see [25, 24], and we plan to further investigate its stability properties in the context of our model.

5 Test for fluid limits (TFL) and stability

This section is devoted to introducing a new methodological strategy to prove stability, that we call *test for fluid limits* (TFL). In Section 6 we use this method to prove all the settings considered in the article. Moreover, we believe that this strategy might be applied to cover other contexts. The idea behind this approach is to obtain the stability of the system as a consequence of the verification of a certain test for the fluid limit, the advantage of course being that this verification is easy to carry out. In this way, we succeed in proving stability without the necessity of explicitly describing the fluid limits, which turns out to be cumbersome in our examples. Before stating Proposition 5.3, the result enclosing this idea, certain prerequisites must be defined.

We introduce the notion of fluid limit following [7]. All our Poisson point processes used to construct our stochastic process are defined in the same probability space $(\Omega, \mathcal{F}, \mathbb{P})$. We fix an almost sure event $\Omega' \in \mathcal{F}$, which can be defined as a set for which the law of large numbers holds for all these Poisson processes. Let $\|\cdot\|$ be the supremum norm in \mathbb{R}^K . For $x = (q, e, c) \in \mathcal{X}$ such that $\|q\| \neq 0$, we define the random function $\bar{Q}^x : [0, \infty) \rightarrow \mathbb{R}^K$ as

$$\bar{Q}^x(t) = \frac{1}{\|q\|} Q^x(\|q\|t).$$

This function is the rescaled queue length process. A sequence $(x^n)_n = ((q^n, e^n, c^n))_n \subseteq \mathcal{X}$ is said to be divergent if $\lim_{n \rightarrow \infty} \|q^n\| = \infty$.

Definition 5.1. *A fluid limit is a (deterministic) function $G : [0, \infty) \rightarrow \mathbb{R}^K$ for which there exist $\omega \in \Omega'$ and a diverging sequence $(x^n)_n$ such that*

$$\lim_{n \rightarrow \infty} \bar{Q}^{x^n}(t)[\omega] = G(t),$$

where the convergence is component-wise and uniformly over compact time subsets.

To a function $G = (G_i)_i : [0, \infty) \rightarrow \mathbb{R}^K$, we associate the max and the argmax functions, which are defined as

$$M(t) = \max_{i \in [K]} G_i(t) \quad \text{and} \quad L(t) = \{i \in [K] : G_i(t) = M(t)\}, \quad t \geq 0.$$

The definition of our test for fluid limits (TFL) follows.

Definition 5.2 (Test for fluid limits (TFL)). *We say that the TFL is passed if there exists $\delta > 0$ such that for every fluid limit G and every pair of times $0 \leq t_1 < t_2$ satisfying*

$$L(t_1) = L(t_2) \text{ and } \min_{i \in L(t_1)} G_i(t) > \max_{i \notin L(t_1)} G_i(t) \text{ for every } t \in [t_1, t_2], \quad (5.1)$$

we have

$$\frac{M(t_2) - M(t_1)}{t_2 - t_1} \leq -\delta. \quad (5.2)$$

In Figure 3, we illustrate condition (5.1).

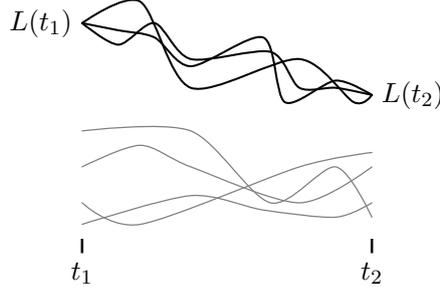


Figure 3: This picture represents condition (5.1). The three leading lines are the graphs of the functions G_i for which i attains the maximum in the extremes of the interval under consideration, $[t_1, t_2]$. The other lines are the graphs of the remaining G_i 's. The two groups of functions are not allowed to intersect among the hole interval $[t_1, t_2]$.

The following result is our methodological tool, that guarantees stability once the test is passed. Its proof is given in Section 5.1 and relies on an inductive procedure that, to the best of our knowledge, is novel.

Proposition 5.3. *If the TFL is passed, then the process is stable.*

5.1 Proof of Proposition 5.3

It is convenient to name the property used in the definition of the TFL.

Property 5.4. *We say that a function $G = (G_i)_i : [0, \infty) \rightarrow \mathbb{R}^K$ satisfies Property 5.4 for $\delta > 0$ if (5.2) holds for every pair of times $0 \leq t_1 < t_2$ satisfying (5.1).*

Under this definition, the TFL is passed if and only if there exists $\delta > 0$ such that every fluid limit satisfies Property 5.4 for $\delta > 0$.

It is convenient to identify another similar property over functions.

Property 5.5. *We say that a function $G = (G_i)_i : [0, \infty) \rightarrow \mathbb{R}^K$ satisfies Property 5.5 for $\delta > 0$ if (5.2) holds for every pair of times $0 \leq t_1 < t_2$ satisfying either*

$$\min_{i \in L(t_1)} G_i(t) > \max_{i \notin L(t_1)} G_i(t) \text{ for every } t \in [t_1, t_2] \quad (5.3)$$

or

$$\min_{i \in L(t_2)} G_i(t) > \max_{i \notin L(t_2)} G_i(t) \text{ for every } t \in [t_1, t_2]. \quad (5.4)$$

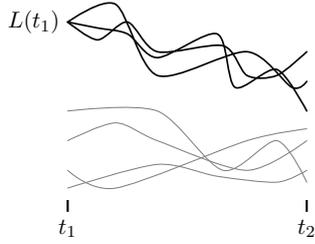


Figure 4: Condition (5.3)

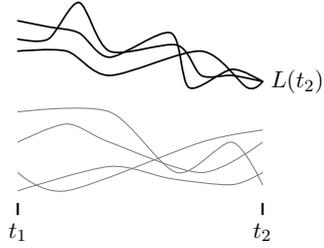


Figure 5: Condition (5.4)

Conditions (5.3) and (5.4) are represented in Figures 4 and 5, respectively. Observe that condition (5.1) in the definition of TFL implies both conditions (5.3) and (5.4). Said in different terms, for a fixed $\delta > 0$, the range of pairs t_1, t_2 over which inequality (5.2) is required to hold in Property 5.5 is larger than in Property 5.4, so every function satisfying Property 5.5 also satisfies Property 5.4. The following result shows that the two properties indeed coincide over Lipschitz functions.

Lemma 5.6. *Let $G : [0, \infty) \rightarrow \mathbb{R}^k$ be a Lipschitz function, and fix $\delta > 0$. Then G satisfies Property 5.4 for δ if and only if it satisfies Property 5.5 for δ .*

Before giving a proof of this lemma, we show how our methodological contribution, Proposition 5.3, follows from it. Suppose that the TFL is passed, that is, there exists $\delta > 0$ such that every fluid limit satisfies Property 5.4 for this $\delta > 0$. Fix now a fluid limit G . By Lemma 5.6, G satisfies Property 5.5 for δ because fluid limits are Lipschitz. Fix t such that $M(t) > 0$ (recall that M is the maximum function associated to G). By continuity, there exists $\varepsilon > 0$ such that (5.3) holds for $t_1 = t$ and any $t_2 \in (t, t + \varepsilon)$, which implies (5.2). The analogous implication holds also to the left, i.e. for $t_2 = t$ and any $t_1 \in (t - \varepsilon, t)$, we have (5.4) and hence (5.2). We have obtained

$$\limsup_{t' \rightarrow t} \frac{M(t) - M(t')}{t - t'} \leq -\delta \quad \text{for every } t \text{ for which } M(t) > 0. \quad (5.5)$$

Since M is Lipschitz, its derivative exists for almost every time, and (5.5) proves that it is bounded by $-\delta$. In other words, the maximum function is Lyapunov for the fluid limit, and the stability of the Markov process under the SLC policy follows, see [19].

Proof of Lemma 5.6

We only prove the sufficiency direction since the necessity direction is trivial. So assume that the Lipschitz function G satisfies Property 5.4 for $\delta > 0$. We will proceed by induction over $m \in \mathbb{N}$, the precise inductive hypothesis being the following one: inequality (5.2) holds for $0 \leq t_1 < t_2$ either when $|L(t_1)| < m$ and (5.3) holds, or when $|L(t_2)| < m$ and (5.4) holds. Observe that the case $m = 1$ holds because in this case Properties 5.4 and 5.5 are equivalent.

Let now $0 \leq t_1 < t_2$ such that $|L(t_1)| = m$ and (5.3) holds. The trick is to define

$$t^* = \max\{t \in [t_1, t_2] : L(t) = L(t_1)\}.$$

Since by assumption inequality (5.2) holds for every pair of times satisfying (5.1), we have

$$\frac{M(t^*) - M(t_1)}{t^* - t_1} \leq -\delta. \quad (5.6)$$

If $t^* = t_2$, we are trivially done. Otherwise, for every $t \in (t^*, t_2)$ we have that $|L(t)| < |L(t_1)|$ due to the validity of condition (5.3) and the very definition of t^* . Hence we can apply (both cases of) the inductive hypothesis to obtain that

$$\limsup_{t' \rightarrow t} \frac{M(t') - M(t)}{t' - t} \leq -\delta \quad \forall t \in (t^*, t_2).$$

From the fundamental theorem of calculus, we get

$$\frac{M(t_2) - M(t^*)}{t_2 - t^*} \leq -\delta. \quad (5.7)$$

Inequalities (5.6) and (5.7) let us obtain the desired inequality (5.2). The second case in which $|L(t_2)| = m$ and (5.4) follows analogously after defining

$$t^* = \min\{t \in [t_1, t_2] : L(t) = L(t_2)\}.$$

6 Proof of the maximum stability region, Theorem 4.1

In this section we prove the sufficiency part of our main result, Theorem 4.1. More precisely, we will prove that the SLC policy is stable under conditions (4.1), (4.2) and (4.3) established for the corresponding settings. To do so, in view of Proposition 5.3, we only need to verify that the corresponding fluid limits pass the TFL. These verifications for Settings II and III are presented in Section 6.1 and Section 6.2, respectively, while we relegate the verification for Setting I to Appendix B.3. The proof that the conditions (4.1), (4.2) and (4.3) are necessary for the existence of a stable policy is postponed to Appendix A.

We use the following standard queue length representation (see [10] for instance). Let $A_i^x \subseteq [0, \infty)$ be the time subset in which queue i is receiving effective service. By effective, we mean that we count only the time when the server is dedicated to this queue and the queue is connected and non-empty. It is formally defined as

$$A_i^x = \{t \in [0, \infty) : C^x(t) = i, E_i^x(t) = 1, Q_i^x(t) > 0\}.$$

For every $i \in [K]$, we call \mathcal{N}_{λ_i} and \mathcal{N}_{μ_i} the counting measures associated to the Poisson point processes respectively used to define the arrivals and departures in queue i . The mentioned queue length representation is given by

$$Q_i^x(t) = q_i + \mathcal{N}_{\lambda_i}([0, t]) - \mathcal{N}_{\mu_i}(A_i^x \cap [0, t]), \quad i \in [K], t \geq 0. \quad (6.1)$$

Fix G to be a fluid limit associated to an element $\omega \in \Omega'$ and a diverging sequence $(x^n)_n$. Since ω is considered fixed, we do not include it in the subsequent notations. An Arzelá-Ascoli argument allows us to prove a fluid version of formula (6.1), namely, for every $i \in [K]$,

$$G_i(t) = G_i(0) + \lambda_i t - \mu_i T_i(t). \quad (6.2)$$

Here T_i is the (uniformly over compacts) limit of the sequence of functions $(\bar{T}_i^{x^n})_n$ defined as

$$\bar{T}_i^{x^n}(t) = \frac{|A_i^{x^n} \cap [0, t \|\bar{q}^n\||}{\|\bar{q}^n\|}, \quad t \geq 0.$$

This sequence represents the scaled versions of the cumulative effective service time.

6.1 TFL verification for Setting II

We first consider Setting II. In order to prove that SLC is stable when (4.2) holds, it is enough to show that the fluid limit corresponding to the policy SLC satisfies the TFL. Recall the definitions of the max and argmax functions M and L associated to the fluid limit G . Take t_1 and t_2 satisfying (5.1). For the sake of notational compactness, we call $L = L(t_1)$, $\tilde{\mu}_i = \mu_i \pi_i(1)$ and $\tilde{\rho}_i = \lambda_i / \tilde{\mu}_i$. We will prove that (5.2), and hence TFL, is satisfied for

$$\delta = \frac{1 - \sum_{i=1}^K \tilde{\rho}_i}{\sum_{i=1}^K \tilde{\mu}_i^{-1}}.$$

Since this is a positive parameter in view of condition (4.2), this implies the stability of the process through Proposition 5.3. The numerator of δ has to be understood as the difference between the capacity of the server and the total effective load; the denominator is a term that appears when normalizing the queue sizes by the effective service rates.

Formula (6.2), in conjunction with assumption (5.1), readily implies

$$\frac{M(t_2) - M(t_1)}{t_2 - t_1} \sum_{i \in L} \tilde{\mu}_i^{-1} = \sum_{i \in L} \frac{G_i(t_2) - G_i(t_1)}{\tilde{\mu}_i(t_2 - t_1)} = \sum_{i \in L} \tilde{\rho}_i - \sum_{i \in L} \frac{T_i(t_2) - T_i(t_1)}{\pi_i(1)(t_2 - t_1)}.$$

We reduced the problem to proving that

$$\sum_{i \in L} \frac{T_i(t_2) - T_i(t_1)}{\pi_i(1)(t_2 - t_1)} = 1 \quad (6.3)$$

because, if true, we have

$$\frac{M(t_2) - M(t_1)}{t_2 - t_1} = \frac{\sum_{i \in L} \tilde{\rho}_i - 1}{\sum_{i \in L} \tilde{\mu}_i^{-1}} \leq -\delta.$$

We explain now why identity (6.3) holds, and we refer for the details to Appendix B.1. The first observation is that, since we are assuming (5.1), the queues in L will have the leading queue sizes during all the considered time interval in the pre-limit. Also, the non-preemptive nature of the policy guarantees that there is always a connected queue at every decision epoch. These two observations imply that, in this regime in which the queues in L are leading, the server will be dedicated precisely to this queue subset L (except possibly during an initial negligible time interval). The LHS of Equation (6.3) captures the proportion of time that the server is assigned to queues in L during the time interval $[t_1, t_2]$. To see this, we note that $T_i(t_2) - T_i(t_1)$ is the amount of time the server is dedicated to i while this queue is connected. When we divide this time by the proportion of time in which queue i is effectively connected during this interval, which is precisely $\pi_i(1)$ as explained in the next paragraph, we get the total amount of time that the server is dedicated to queue i . Thus, $\frac{T_i(t_2) - T_i(t_1)}{\pi_i(1)(t_2 - t_1)}$ represents the proportion of time that the server is dedicated to i . Since, as just explained, the server is fully dedicated to queues in L , we can conclude that these fractions sum up to 1, yielding (6.3).

We now explain why the proportion described above is given by $\pi_i(1)$. Consider the stochastic process that encodes the state of the i -th environment only during the time the server is dedicated to it. This is a Markov process with state-space $\{0, 1\}$ that transitions from disconnected to connected at rate λ'_i , and from connected to disconnected at rate μ'_i . Its invariant distribution is precisely π_i , and by the ergodic theorem $\pi_i(1)$ is the desired proportion. For clarity, it is important to note that, if a departure occurs in queue i , the state of its environment, 1, remains unchanged. Indeed, since the policy is in the family \mathcal{P} , even if the server is dedicated to other queues for a period of time after this departure, the i -th queue will necessarily be connected again when the server is re-dedicated to it.

6.2 TFL verification for Setting III

Define $\hat{\mu}_i = \theta_i(\gamma)\mu_i$ and $\hat{\rho}_i = \lambda_i/\hat{\mu}_i$ for notational compactness. As in the previous case, but replacing $\pi_i(1)$ by $\theta_i(\gamma)$, we get

$$\frac{M(t_2) - M(t_1)}{t_2 - t_1} \sum_{i \in L} \hat{\mu}_i^{-1} = \sum_{i \in L} \hat{\rho}_i - \sum_{i \in L} \frac{T_i(t_2) - T_i(t_1)}{\theta_i(\gamma)(t_2 - t_1)}.$$

We will prove in Appendix B.1 that

$$\sum_{i \in L} \frac{T_i(t_2) - T_i(t_1)}{\theta_i(\gamma)(t_2 - t_1)} = 1 - \pi_L^0, \quad (6.4)$$

which implies the desired inequality (5.2) for

$$\delta = \frac{\min\{1 - \pi_L^0 - \sum_{i \in L} \hat{\rho}_i : L \subseteq [K], L \neq \emptyset\}}{\sum_{i=1}^K \hat{\mu}_i^{-1}}.$$

The numerator of δ represents the minimum over subsets L of the difference between the maximum capacity that the server can dedicate to the subset and the total effective load in question.

Identity (6.4) has a similar explanation as (6.3), with some differences that we now outline. For further details we refer to Appendix B.2. The first difference is that the coefficient equals $\theta_i(\gamma)$ and not $\pi_i(1)$. As before, we define the Markov process encoding the state of the i -th environment during the time the server is dedicated to queue i , which in this case transitions from disconnected to connected at rate $\lambda'_i + \gamma$, and from connected to disconnected at rate μ'_i . The reason is akin to what was mentioned earlier, with the additional matter that we pass from disconnected to connected also when the exponential clock of intensity γ rings. Hence, by the ergodic theorem, $\theta_i(\gamma) = \frac{\lambda'_i + \gamma}{\mu'_i + \lambda'_i + \gamma}$ is the asymptotic proportion of time this process is connected as expected.

The second difference is that the RHS of (6.4) is $1 - \pi_L^0$ and not 1 as in the previous case. To understand this consider two consecutive γ -marks $s_1, s_2 \in (t_1^n, t_2^n]$. In an interval $[s_1, s_2)$ the server is either dedicated to a fixed queue, or is dedicated to \emptyset . Since we are under Setting III and consider SLC, a queue i in L is always served, except when all queues in L are disconnected at time s_1 . The latter happens with probability π_L^0 . Therefore, $1 - \pi_L^0$ is the asymptotic proportion of this type of intervals (those within $(t_1^n, t_2^n]$ that are delimited by γ -marks) for which queues in L are served. Since the LHS of (6.4) is the proportion of time that the server is assigned to queues in L , this should hence be equal to $1 - \pi_L^0$.

7 Numerical evaluation of stability regions

In this section we provide some numerical examples of the comparison results we described in Section 4. In particular we want to assess the impact of γ on the reduction in the stability region, $\text{Red}_{\text{III}}(\gamma)$, for which a closed-form expression is given in (4.5).

In Figure 6 (*left*), we plot the reduction in the maximum stability region when decision are at γ epochs, i.e. $\text{Red}_{\text{III}}(\gamma)$, as a function of the decision rate γ . The difference between the two lines lies in the speed of the environments, which is 10 times faster in the *dash* line. As expected from (4.5), the reduction of the stability decreases as γ increases and converges to zero. We further observe that the *solid* line decreases much faster. The intuition for this is that the state of connectivity of the queues changes less fast when $\alpha = 1$, and hence, a smaller rate of decisions is already sufficient to get a similar stability reduction as for the case where the connectivity changes faster, $\alpha = 10$.

In Figure 6 (*right*) we illustrate the result of Corollary 4.3. We take a symmetric situation with $\lambda'_i = \mu'_i = 1$ and $\rho_1 = \rho_2$. From Theorem 4.1 we obtain that in order to be stable in Setting I, the load should satisfy $\rho_i \in [0, (1 - \pi_1(0)\pi_2(0))/2)$. In Figure 6 (*right*) we plot γ_0 as a function of ρ_i . We recall that γ_0 , defined in Corollary 4.3, denotes the minimal value of γ such that the system is stable in Setting III. We observe that the minimal value of γ_0 remains relatively small until ρ_i gets very close to the stability border of MSR_{I} , which shows a sharp phase transition: stabilizing the system in terms of decision rate is relatively cheap until we are very close to the frontier.

In Figure 7 (*left*) we illustrate the result of Corollary 4.5, that is, we plot the minimum rate γ_1 such that for any $\gamma \geq \gamma_1$, the reduction in stability for Setting III is less than the tolerable reduction

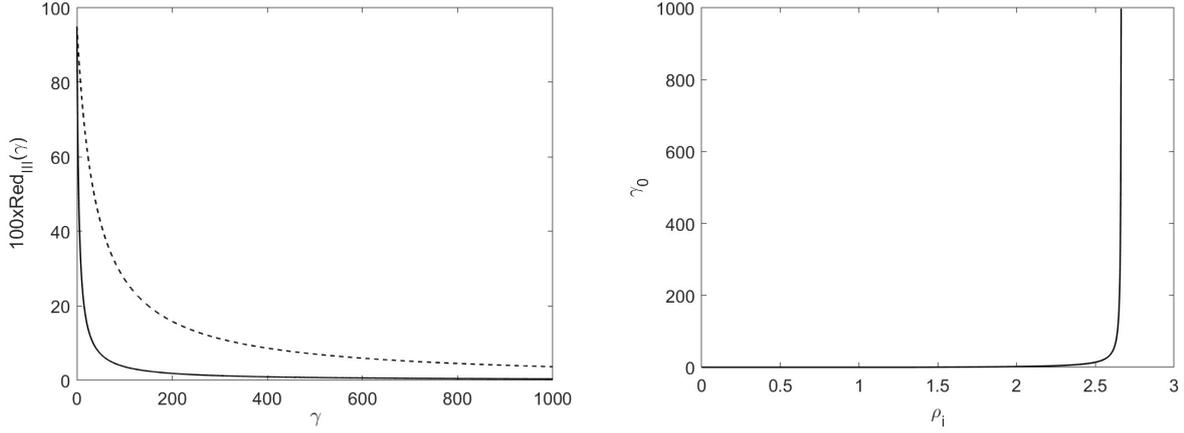


Figure 6: (left) $\text{Red}_{\text{III}}(\gamma)$ as a function of the parameter γ , with $\lambda'_1 = \alpha 0.8, \lambda'_2 = \alpha 3, \mu'_1 = 0.2\alpha$ and $\mu'_2 = \alpha$. We set $\alpha = 1$ for the *solid* line and $\alpha = 10$ for the *dashed* line. (right) Minimal γ_0 as a function of ρ_i , with $\lambda'_i = \mu'_i = 1$ and $\rho_1 = \rho_2$.

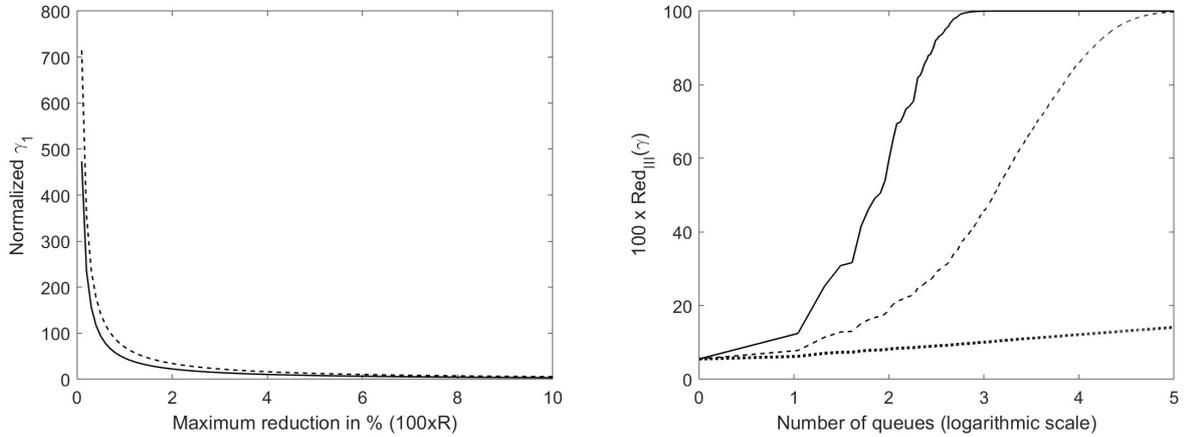


Figure 7: (left) The normalized value $\frac{\gamma_1 - U}{U}$ as a function of the tolerable stability reduction R . The parameters from Figure 6 (left) are used here. (right) $\text{Red}_{\text{III}}(\gamma)$ versus the number of queues, where $\gamma = 5$ for the *solid* line, $\gamma = 5\sqrt{K}$ for the *dashed* curve and $\gamma = 5K$ for the *dotted* line.

given by R . To find γ_1 , we simply solve the 2nd degree polynomial obtained from $\text{Red}_{\text{III}}(\gamma) = (1 - \theta_1(\gamma)\theta_2(\gamma)) = R$. Again, we consider two set of parameters, where the difference is that the rates of connectivity are scaled. In order to compare the values of γ_1 , we scale them by $U = \lambda'_1 + \lambda'_2 + \mu'_1 + \mu'_2$ and plot $(\gamma_1 - U)/U$. Firstly, we see that both curves overlap quite closely, despite the fact that the rates of the environments for the dashed line are 10 times larger. Secondly, we observe that if we want to make the stability reduction to be smaller than, say 1%, the parameter γ_1 needs to be around 50 times larger than the sum of the transition rates, U .

In Figure 7 (right) we plot the reduction in stability as a function of the number of queues, K . The parameters of the queue, λ'_i and μ'_i , are chosen randomly from a uniform distribution on $[0, 1]$. In the *solid* curve, the value of γ is kept constant, and we see that the reduction tends to 100%, as could also be seen directly from Equation (4.5). In the *dash* line, we scale the value of γ with the square root of the number of servers, and we observe that, even though more slowly, the reduction still converges to 100%. In the *dotted* line, we scale γ linearly with the number of servers and we observe that the stability reduction tends to some positive value strictly smaller than 1. It would be

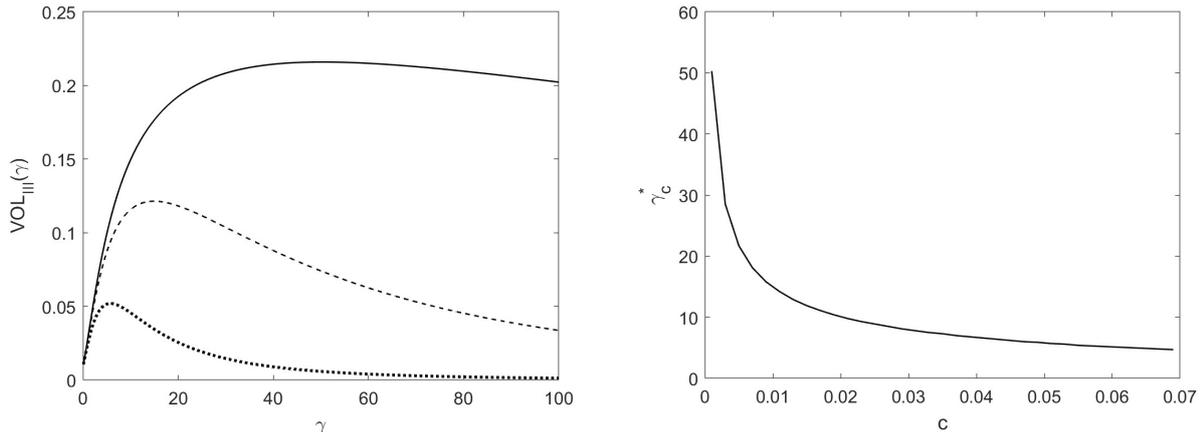


Figure 8: We set $\lambda'_1 = 1$, $\lambda'_2 = 2$, $\lambda'_3 = 3$, $\mu'_1 = 3$ and $\mu'_2 = 2$ and $\mu'_3 = 1$. (left) $VOL_{III}(\gamma)$ (as determined by the inequalities (4.6)) as a function of γ . The inactivity duration c equals $c = 0.001$ for the *solid* line, $c = 0.01$ for the *dashed* line, and $c = 0.05$ for the *dotted* line. (right) Optimal decision rate, γ_c^* , as a function of c .

interesting to show that indeed scaling the decision rate γ linearly with the number of servers yields a constant stability reduction asymptotically.

In Figure 8, we illustrate the result of Corollary 4.6 where the stability region is determined for a system where each decision epoch causes an inactivity period of duration c . We set $K = 3$ and in Figure 8 (left) we plot $VOL_{III}(\gamma)$ (can be calculated from Equation 4.6) as a function of γ . We observe the existence of an optimal rate γ_c^* that maximizes the stability region and hence strikes the right balance between a high frequency of taking actions and the overheads induced by them. In Figure 8 (right) we plot γ_c^* as a function of c . We observe that γ_c^* is decreasing in c . Indeed, the larger c , the more harmful it becomes to increase the frequency of actions.

8 Conclusions

Building on an original analysis of fluid limits of one server multi-class system in a random environment, our results reveal the crucial impact of decision epochs on the stability properties of a natural class \mathcal{P} of policies. We conclude mentioning a few research avenues that stem out of our work. Exploring the maximal stability region for policies outside \mathcal{P} remains an open problem (see Section 4.6). Besides, exploring the impact of decision epochs on the stability region opens avenues for further investigation, for instance in stochastic networks with more complex topologies and broader statistical assumptions.

More generally, our approach finds potential application for the study of continuous-time Partially Observable Markov Decision Processes (POMDP) with constrained action epochs and their model-free counterpart in reinforcement learning, which currently gathers considerable attention. Finally, we also mention that the methodological contribution we present on fluid limits, could be promising for examining the stability of various other models.

References

- [1] M. Andrews, K. Kumaran, K. Ramanan, A. Stolyar, R. Vijayakumar, and P. Whiting. Scheduling in a queuing system with asynchronously varying service rates. *Probability in the Engineering and Informational Sciences*, 18(2):191–217, 2004.

- [2] 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.
- [3] F. Baccelli and A. M. Makowski. Stability and bounds for single server queues in random environment. *Communications in Statistics. Stochastic Models*, 2(2):281–291, 1986.
- [4] N. Bambos and G. Michailidis. Queueing and scheduling in random environments. *Advances in Applied Probability*, 36(1):293–317, 2004.
- [5] Nicholas Bambos and George Michailidis. Queueing networks of random link topology: stationary dynamics of maximal throughput schedules. *Queueing Syst.*, 50(1):5–52, 2005.
- [6] S.C. Borst. User-level performance of channel-aware scheduling algorithms in wireless data networks. *IEEE/ACM Transactions on Networking*, 13(3):636–647, 2005.
- [7] M. Bramson. *Stability of queueing networks*, volume 1950. Springer, Berlin, Berlin, 2008.
- [8] Amarjit Budhiraja, Arka Ghosh, and Xin Liu. Scheduling control for markov-modulated single-server multiclass queueing systems in heavy traffic. *Queueing Systems*, 78(1):57–97, 2014.
- [9] G. D Celik, L. B. Le, and E. Modiano. Scheduling in parallel queues with randomly varying connectivity and switchover delay. In *2011 Proceedings IEEE INFOCOM*, pages 316–320. IEEE, 2011.
- [10] J. G. Dai. On positive Harris recurrence of multiclass queueing networks: a unified approach via fluid limit models. *Ann. Appl. Probab.*, 5(1):49–77, 1995.
- [11] B. D’Auria. $m/m/\infty$ queues in semi-markovian random environment. *Queueing Systems*, (58):221–237, 2008.
- [12] Antonis Dimakis and Jean Walrand. Sufficient conditions for stability of longest-queue-first scheduling: second-order properties using fluid limits. *Advances in Applied Probability*, 38(2):505–521, 2006.
- [13] H. T Dinh, C. Lee, D. Niyato, and P. Wang. A survey of mobile cloud computing: architecture, applications, and approaches. *Wireless communications and mobile computing*, 13(18):1587–1611, 2013.
- [14] Santiago Duran and Ina Maria Maaïke Verloop. Asymptotic Optimal Control of Markov-Modulated Restless Bandits. In *ACM Sigmetrics 2018*, Irvine, United States, June 2018.
- [15] M. Khabbaz and C. M. Assi. Modelling and analysis of a novel deadline-aware scheduling scheme for cloud computing data centers. *IEEE Transactions on Cloud Computing*, 6(1):141–155, 2015.
- [16] L. Kleinrock. *Queueing Systems, vol. 1*. John Wiley and Sons, 1976.
- [17] L. B. Le, E. Modiano, C. Joo, and N. B. Shroff. Longest-queue-first scheduling under the sinr interference model. In *ACM MobiHoc*, 2010.
- [18] Z. Peng, D. Cui, J. Zuo, Q. Li, B. Xu, and W. Lin. Random task scheduling scheme based on reinforcement learning in cloud computing. *Cluster computing*, 18:1595–1607, 2015.
- [19] P. Robert. *Stochastic networks and queues*, volume 52 of *Applications of Mathematics (New York)*. Springer-Verlag, Berlin, french edition, 2003. Stochastic Modelling and Applied Probability.

- [20] S. Shakkottai and A. L. Stolyar. Scheduling for multiple flows sharing a time-varying channel: The exponential rule. *Translations of the American Mathematical Society-Series 2*, 207:185–202, 2002.
- [21] R. Srikant and L. Ying. *Communication Networks: An Optimization, Control, and Stochastic Networks Perspective*. Cambridge University Press, 2013.
- [22] A.L. Stolyar. Maxweight scheduling in a generalized switch: State space collapse and workload minimization in heavy traffic. *Annals of Applied Probability*, 14(1):1–53, 2004.
- [23] L. Tassiulas and A. Ephremides. Dynamic server allocation to parallel queues with randomly varying connectivity. *IEEE Transactions on Information Theory*, 39(2):466–478, 1993.
- [24] I.M. Verloop. Asymptotically optimal priority policies for indexable and non-indexable restless bandits. *Annals of Applied Probability*, 26(4):1947–1995, 2016.
- [25] Richard R Weber and Gideon Weiss. On an index policy for restless bandits. *Journal of applied probability*, pages 637–648, 1990.
- [26] P. Whittle. Restless bandits: Activity allocation in a changing world. *Journal of Applied Probability*, 25:287–298, 1988.

A Necessity proof of Theorem 4.1

We prove now that the conditions given in Theorem 4.1 are necessary.

Setting I

For this setting, this result is not new (see [5] for instance), but nevertheless we give a proof for completeness. Suppose that condition (4.1) is not fulfilled, namely, that there exists a non-empty subset of queues $L \subseteq [K]$ such that

$$\sum_{i \in L} \rho_i > 1 - \pi_L^0.$$

Under this condition, we will prove that any policy is unstable, the idea being that, even in the extreme scenario in which all the service is dedicated to the subset L , tasks will necessarily accumulate in these queues.

From formula (6.1), for an arbitrary initial condition x , we get

$$\sum_{i \in L} \frac{Q_i^x(t)}{t\mu_i} = \sum_{i \in L} \frac{q_i}{t\mu_i} + \sum_{i \in L} \frac{\mathcal{N}_{\lambda_i}([0, t])}{t\mu_i} - \sum_{i \in L} \frac{\mathcal{N}_{\mu_i}(A_i^x \cap [0, t])}{t\mu_i}. \quad (\text{A.1})$$

The last sum can be controlled as follows:

$$\begin{aligned} \sum_{i \in L} \frac{\mathcal{N}_{\mu_i}(A_i^x \cap [0, t])}{t\mu_i} &\leq \sum_{i \in L} \frac{\mathcal{N}_{\mu_i}(\{s \in [0, t] : C^x(s) = i, E_i^x(s) = 1\})}{t\mu_i} \\ &= \frac{1}{t} \sum_{i \in L} |\{s \in [0, t] : C^x(s) = i, E_i^x(s) = 1\}| + R_t \\ &\leq \frac{1}{t} \left| \bigcup_{i \in L} \{s \in [0, t] : E_i^x(s) = 1\} \right| + R_t. \end{aligned} \quad (\text{A.2})$$

In the first inequality, we simply neglected the per-queue unbusy periods, i.e., the periods in which the queue is empty. The error R_t is defined for the equality to hold. The second inequality involves two steps: first, we use that the sets appearing in the sum in (A.2) are disjoint; secondly, as anticipated, we bound by the most beneficial case in which all the service is dedicated to the queues in L . Since R_t goes to zero as $t \rightarrow \infty$ —due to the law of large numbers—, the last expression converges almost surely to $1 - \pi_L^0$. Coming back to (A.1), and using that the second sum in the RHS of (A.1) converges to $\sum_{i \in L} \rho_i$, we obtain

$$\liminf_{t \rightarrow \infty} \sum_{i \in L} \frac{Q_i^x(t)}{t\mu_i} \geq \sum_{i \in L} \rho_i - (1 - \pi_L^0) \quad a.s.$$

Since by assumption the RHS in this inequality is strictly positive, we can conclude that

$$\liminf_{t \rightarrow \infty} \sum_{i \in L} \frac{Q_i(t)}{\mu_i} = +\infty \quad a.s.,$$

and hence the process is unstable.

Setting II

Suppose that condition (4.2) is not satisfied, i.e., suppose that

$$\sum_{i=1}^K \frac{\rho_i}{\pi_i(i)} > 1.$$

Under this assumption, we will prove that any policy is unstable.

Again from formula (6.1),

$$\sum_{i=1}^K \frac{Q_i(t)}{t\pi_i(1)\mu_i} = \sum_{i=1}^K \frac{q_i}{t\pi_i(1)\mu_i} + \sum_{i=1}^K \frac{\mathcal{N}_{\lambda_i}([0, t])}{t\pi_i(1)\mu_i} - \sum_{i=1}^K \frac{\mathcal{N}_{\mu_i}(A_i^x \cap [0, t])}{t\pi_i(1)\mu_i}. \quad (\text{A.3})$$

We can deal with the last sum as follows:

$$\begin{aligned} \sum_{i=1}^K \frac{\mathcal{N}_{\mu_i}(A_i^x \cap [0, t])}{t\pi_i(1)\mu_i} &\approx \frac{1}{t} \sum_{i=1}^K \frac{|\{s \in [0, t] : C^x(s) = i, E_i^x(s) = 1\}|}{\pi_i(1)} \\ &\approx \frac{1}{t} \sum_{i=1}^K |\{s \in [0, t] : C^x(s) = i\}|. \end{aligned} \quad (\text{A.4})$$

The symbol \approx means ‘less or equal with an error that vanishes as $t \rightarrow \infty$ ’. In the first line, we neglected the unbusy periods and used the law of large numbers, exactly as we did right before in the necessity proof for Setting I. To justify the second line, we refer to the argument that we used at the end of the test verification for the same Setting II, in Section 6. In that argument, we used that $\pi_i(1)$ is the asymptotic proportion of time in which a certain queue i is connected within the time it is receiving service. In the case of a general policy, $\pi_i(1)$ instead works as an upper bound for such a proportion because it may happen that, at a decision epoch, we serve a queue that is disconnected. If this happens, the time interval between such epoch and the moment in which the queue under service connects, is a disconnected time interval that is not counted in the SLC case. Using that the expression in (A.6) is bounded by 1, we obtain

$$\limsup_{t \rightarrow \infty} \sum_{i=1}^K \frac{\mathcal{N}_{\mu_i}(A_i^x \cap [0, t])}{t\pi_i(1)\mu_i} \leq 1 \quad a.s.$$

Substituting in (A.3), we get

$$\liminf_{t \rightarrow \infty} \sum_{i=1}^K \frac{Q_i(t)}{t\mu_i\pi_i(1)} \geq \sum_{i=1}^K \frac{\rho_i}{\pi_i(i)} - 1 \quad a.s.,$$

which let us conclude.

Setting III

Suppose that

$$\sum_{i \in L} \frac{\rho_i}{\theta_i(\gamma)} > 1 - \pi_L^0 \quad (\text{A.5})$$

for some non-empty subset $L \subseteq [K]$. Unlike the other settings, in Setting III we can only prove that any policy in the family \mathcal{P} is unstable. This is related with the discussion given in 4.6, in which we mention that, even if (A.5) holds, there might be stable policies outside the family \mathcal{P} .

The proof in this setting has the same spirit than in the previous ones. In this case we have again formula (A.3), but with $\theta_i(\gamma)$ instead of $\pi_i(1)$, and with the sums running only over the queues in L . We control the sum concerning the services as

$$\begin{aligned} \sum_{i \in L} \frac{\mathcal{N}_{\mu_i}(A_i^x \cap [0, t])}{t\theta_i(\gamma)\mu_i} &\approx \frac{1}{t} \sum_{i \in L} \frac{|\{s \in [0, t] : C^x(s) = i, E_i^x(s) = 1\}|}{\theta_i(\gamma)} \\ &\approx \frac{1}{t} \sum_{i \in L} |\{s \in [0, t] : C^x(s) = i\}| \\ &\approx 1 - \pi_L^0. \end{aligned} \quad (\text{A.6})$$

Of course, the symbol \approx means ‘equal with an error that vanishes as $t \rightarrow \infty$ ’. In the first line, we neglected the per-queue unbusy periods, as usual. In the second line, we used that $\theta_i(\gamma)$ is the proportion of connected periods within the time that we are serving queue i . We emphasize that this is an inherent property of the family \mathcal{P} and not of the SLC policy, and this is also the reason why, in this line, we have an \approx instead of an \lesssim as we had in the Setting II. The last line is due to the fact that the proportion of time intervals separated by consecutive γ -marks in which we serve a queue in L is at most $1 - \pi_L^0$ because, since the policy is in the family \mathcal{P} , at decision epochs we cannot select a queue that is disconnected. Importantly, this last step would fail if we considered policies outside \mathcal{P} .

B Remaining proofs from the TFL verifications

B.1 Proof of Equations (6.3)

This identity follows from the convergence

$$\lim_{n \rightarrow \infty} \sum_{i \in L} \frac{\bar{T}_i^{x^n}(t_2) - \bar{T}_i^{x^n}(t_1)}{\pi_i(1)(t_2 - t_1)} = 1 \quad (\text{B.1})$$

simply because, by definition, $\bar{T}_i^{x^n}$ approximates T_i . Let $t_1^n = \|q^n\|t_1$ and $t_2^n = \|q^n\|t_2$ represent the microscopic versions of t_1 and t_2 . Let also τ_n be the first time after t_1^n a queue in L receives service, formally defined as

$$\tau_n = \inf\{t \geq t_1^n : C^{x^n}(t) \in L\}. \quad (\text{B.2})$$

Considering that it is sufficient for one of the queues in L to be connected in an decision epoch for the SLC policy to start serving this group of queues, it is not difficult to believe that

$$\lim_{n \rightarrow \infty} \frac{\tau_n}{\|q^n\|} = t_1. \quad (\text{B.3})$$

This is proven at the end of the section.

Convergence (B.1) follows from the following sequence of steps:

$$\begin{aligned} \sum_{i \in L} \frac{\bar{T}_i^{x^n}(t_2) - \bar{T}_i^{x^n}(t_1)}{\pi_i(1)(t_2 - t_1)} &= \sum_{i \in L} \frac{|A_i^{x^n} \cap (t_1^n, t_2^n]|}{\pi_i(1)(t_2^n - t_1^n)} \\ &\approx \sum_{i \in L} \frac{|A_i^{x^n} \cap (\tau_n, t_2^n]|}{\pi_i(1)(t_2^n - \tau_n)} \\ &= \sum_{i \in L} \frac{|\{t \in (\tau_n, t_2^n] : C^{x^n}(t) = i, E_i^{x^n}(t) = 1, Q_i^{x^n}(t) > 0\}|}{\pi_i(1)(t_2^n - \tau_n)} \\ &\approx \sum_{i \in L} \frac{|\{t \in (\tau_n, t_2^n] : C^{x^n}(t) = i, E_i^{x^n}(t) = 1\}|}{\pi_i(1)(t_2^n - \tau_n)} \\ &\approx \sum_{i \in L} \frac{|\{t \in (\tau_n, t_2^n] : C^{x^n}(t) = i\}|}{t_2^n - \tau_n} \xrightarrow{n \rightarrow \infty} 1. \end{aligned}$$

The symbol \approx means ‘equal with an error that vanishes as $n \rightarrow \infty$ ’. The first and second identities ((B.1) and (B.1)) are respectively because of the definitions of $\bar{T}_i^{x^n}$ and $A_i^{x^n}$. The first approximation (B.1) is due to (B.3). The validity of (5.1) implies that its microscopic version

$$\min_{i \in L} \bar{Q}_i^{x^n}(t) > \max_{i \notin L} \bar{Q}_i^{x^n}(t) \quad \forall t \in [t_1, t_2] \quad (\text{B.4})$$

holds for n large enough, which explains the second approximation in which we simply neglect the condition of having a positive queue length. The convergence to 1 stated in the last line also uses the asymptotic validity of (B.4). Indeed, for large values of n , the SLC policy will dedicate service only to the queues in L during the time interval $(\tau_n, t_2^n]$ because, on the one hand, the leading queue is necessarily in L and, on the other hand, the queue under service is connected at decision epochs as explained before. We recall that the last approximation was already explained towards the end of Section 6.1.

While starting with x^n as initial condition, let $([a^n(m), b^n(m)))_{m \in \mathbb{N}}$ be the sequence of intervals after $t_1 \|q^n\|$ inside which all the environments are connected. We now prove convergence (B.2). Let $A_i^n(m)$ be the event defined as follows: in the time interval $[a^n(m), b^n(m))$, there are exactly one λ_i -mark and exactly one μ_i -mark, and the λ_i -mark comes before the μ_i -mark. Let $A^n(m) = \bigcap_{i \in [K]} A_i^n(m)$. For n large enough such that (B.4) holds, the occurrence of $A^n(m)$ for an m such that $b^n(m) \leq t_2 \|q^n\|$ implies that $C^{x^n}(b^n(m)) \in L$. Indeed, under these conditions, we can guarantee that a decision epoch occurs inside $[a^n(m), b^n(m))$, epoch at which we will pass to serve the maximum queue (that is in L) because all the queues are connected. In conclusion, if $A^n(m)$ occurs then $\tau^n \leq b^n(m)$, or, in other terms,

$$\tau^n \leq \min\{b^n(m) : m \in \mathbb{N}, b^n(m) \leq t_2 \|q^n\|, A^n(m) \text{ occurs}\}.$$

We can conclude from the fact that the occurrences of the events $\{A^n(m) : m \in \mathbb{N}\}$ represent independent trials of positive probability.

B.2 Proof of (6.4)

It follows from the following steps:

$$\begin{aligned} \sum_{i \in L} \frac{\bar{T}_i^{x^n}(t_2) - \bar{T}_i^{x^n}(t_1)}{\theta_i(\gamma)(t_2 - t_1)} &\approx \sum_{i \in L} \frac{|\{t \in (t_1^n, t_2^n] : C^{x^n}(t) = i, E_i^{x^n}(t) = 1\}|}{\theta_i(\gamma)(t_2^n - t_1^n)} \\ &\approx \sum_{i \in L} \frac{|\{t \in (t_1^n, t_2^n] : C^{x^n}(t) = i\}|}{t_2^n - t_1^n} \xrightarrow{n \rightarrow \infty} 1 - \pi_L^0. \end{aligned} \quad (\text{B.5})$$

The first approximation follows as in Section B.1 but without the necessity of introducing the stopping time τ_n . We highlight that (5.1) is required in this step. The second approximation and the convergence were already explained in Section 6.2.

B.3 Test verification for $J = I$

We will verify the TFL for

$$\delta = \frac{\min\{1 - \pi_L^0 - \sum_{i \in L} \rho_i : L \subseteq [K], L \neq \emptyset\}}{\sum_{i=1}^K \mu_i^{-1}},$$

which is a positive quantity due to condition (4.1). Let $t_1 < t_2$ satisfying (5.1), and call $L = L(t_1) = L(t_2)$. As in the other settings, we have

$$\frac{M(t_2) - M(t_1)}{t_2 - t_1} \sum_{i \in L} \mu_i^{-1} = \sum_{i \in L} \rho_i - \sum_{i \in L} \frac{T_i(t_2) - T_i(t_1)}{t_2 - t_1},$$

and the proof is now reduced to proving that

$$\sum_{i \in L} \frac{T_i(t_2) - T_i(t_1)}{t_2 - t_1} = 1 - \pi_L^0.$$

As before, condition (5.1) let us obtain this identity through the following steps:

$$\begin{aligned} \sum_{i \in L} \frac{\bar{T}_i^{x^n}(t_2) - \bar{T}_i^{x^n}(t_1)}{\mu_i(t_2 - t_1)} &\approx \frac{\sum_{i \in L} |\{t \in (t_1^n, t_2^n] : C^{x^n}(t) = i, E_i^{x^n}(t) = 1\}|}{t_2^n - t_1^n} \\ &\approx \frac{|\{t \in (t_1^n, t_2^n] : E_i^{x^n}(t) = 1 \text{ for some } i \in L\}|}{t_2^n - t_1^n} \xrightarrow{n \rightarrow \infty} 1 - \pi_L^0. \end{aligned}$$

C Additional proofs

C.1 Proof of Corollary 4.4

We first note that vertices characterizing both MSR_I and $\text{MSR}_{III}^{\mathcal{P}}(\gamma)$ are related. Indeed, if (x_1, x_2, \dots, x_K) is a vertex of MSR_I , it then follows that $(\theta_1(\gamma)x_1, \theta_2(\gamma)x_2, \dots, \theta_K(\gamma)x_K)$ is a vertex of $\text{MSR}_{III}^{\mathcal{P}}(\gamma)$. To see this it suffices to observe that with the change of variable $\tilde{\rho}_i = \rho_i/\theta_i(\gamma)$, the inequalities of Setting III, see Equation (4.3), coincide with those of Setting I, see Equation (4.1).

This now allows us to relate their volumes. The volume is a measure of the "size" of the convex set and if we scale each dimension by a factor $\theta_i(\gamma)$, the volume scales by the product $\theta_1(\gamma) \times \dots \times \theta_K(\gamma)$. In other words, VOL_I and $\text{VOL}_{III}(\gamma)$ satisfy the relation $\text{VOL}_{III}(\gamma) = \theta_1(\gamma)\theta_2(\gamma) \dots \theta_K(\gamma)\text{VOL}_I$, which directly yields the expression for $\text{Red}_{III}(\gamma)$ in Equation (4.5).

C.2 Proof of Corollary 4.6

We can follow the exact same steps of the proof of stability of the system without delays. In equation (B.2), the limit is replaced by $\frac{1/\gamma}{\frac{1}{\gamma} + c}(1 - \pi_L^0)$.

C.3 Proof of Proposition 4.7

The proof is very similar to the test verification for $J = III$ given in Section 6. With the new policy, the RHS of (B.5) need to be decomposed as

$$\begin{aligned} &\sum_{i \in L} \frac{|\{t \in (t_1^n, t_2^n] : C^{x^n}(t) = i, E_i^{x^n}(t) = 1, t \in B_n\}|}{\theta_i(\gamma)(t_2^n - t_1^n)} \\ &+ \sum_{i \in L} \frac{|\{t \in (t_1^n, t_2^n] : C^{x^n}(t) = i, E_i^{x^n}(t) = 1, t \notin B_n\}|}{\theta_i(\gamma)(t_2^n - t_1^n)}. \end{aligned}$$

Here B_n is the union of the intervals delimited by consecutive γ -marks for which all the queues are disconnected at the beginning of the interval. The second sum is controlled as we did before, with the only difference that the concerning tailor-made Markov process only considers the time outside B_n . To control the first sum, we need to define a similar process but that only records the time inside B_n . For the same reasons than before, the asymptotic proportion of time that such a new process is connected is

$$\phi_i(\gamma) = \frac{\lambda'_i}{\lambda'_i + \mu'_i + \gamma}.$$

If we define

$$\varepsilon(\gamma) = \min_{i \in [K]} \frac{\phi_i(\gamma)}{\theta_i(\gamma)} = \min_{i \in [K]} \frac{\lambda'_i}{\gamma + \lambda_i},$$

then the second sum is asymptotically larger or equal than $\varepsilon(\gamma)\pi_L^0$. Repeating the same steps than before, we obtain that the region

$$\sum_{i \in L} \frac{\rho_i}{\theta_i(\gamma)} < 1 - \pi_L^0(1 - \varepsilon(\gamma)) \quad \forall L \subseteq [K], L \neq \emptyset,$$

which of course strictly contains $\text{MSR}_{\text{III}}^{\mathcal{P}}(\gamma)$, is contained in the stability region of the policy at issue.

To prove the second inequality, we observe that

$$\varepsilon'(\gamma) = \min_{i \in [K]} \phi_i(\gamma)$$

gives a lower bound for the proportion of capacity wasted when the γ -decision epochs are imposed, whatever the policy is and whichever queue is receiving service. Fix a queue i , take ρ_i such that

$$1 - \pi_i(0) - \varepsilon'(\gamma) < \rho_i < 1 - \pi_i(0),$$

and chose the rest of the ρ_j 's to be inside MSR_{I} . Under these choices, we are of course inside the maximal stability region for Setting I, but at the same time tasks in queue i accumulate in Setting III, and hence we are outside $\text{MSR}_{\text{III}}(\gamma)$ as desired.