Distributed Proportional Fair Load Balancing in Heterogenous Systems

Se-Young Yun
MSR-INRIA
Paris, France
seyoung.yun@inria.fr

Alexandre Proutiere*
KTH
Stockholm, Sweden
alepro@kth.se

ABSTRACT

We consider the problem of distributed load balancing in heterogenous parallel server systems, where the service rate achieved by a user at a server depends on both the user and the server. Such heterogeneity typically arises in wireless networks (e.g., servers may represent frequency bands, and the service rate of a user varies across bands). We assume that each server equally shares in time its capacity among users allocated to it. Users initially attach to an arbitrary server, but at random instants of time, they probe the load at a new server and migrate there if this improves their service rate. The dynamics under this distributed load balancing scheme, referred to as Random Local Search (RLS), may be interpreted as those generated by strategic players updating their strategy in a load balancing game. In closed systems, where the user population is fixed, we show that this game has pure Nash Equilibriums (NEs), and that these equilibriums get close to a Proportionally Fair (PF) allocation of users to servers when the user population grows large. We provide an anytime upper bound of the gap between the allocation under RLS and the PF allocation. In open systems, where users randomly enter the system and leave upon service completion, we establish that the RLS algorithm stabilizes the system whenever this it at all possible under centralized load balancing schemes, i.e., it is throughput-optimal. The proof of this result relies on a novel Lyapounov analysis that captures the dynamics due to both users' migration and their arrivals and departures. To our knowledge, the RLS algorithm constitutes the first fully distributed and throughput-optimal load balancing scheme in heterogenous parallel server systems. We extend our analysis to various scenarios, e.g. to cases where users can be simultaneously served by several servers. Finally we illustrate through numerical experiments the efficiency of the RLS algorithm.

1. INTRODUCTION

Load balancing is an essential component in computer systems; it ensures high resource utilization and guarantees satisfactory quality of service. Traditionally, load balancing is performed when tasks arrive in the system: a task is assigned to a carefully selected server, and stays there until service completion. In this paper, we investigate systems where tasks are initially assigned to arbitrary servers, but may be re-assigned to other servers during their services. An additional important property of the systems considered here lies in their *heterogeneity*: the service speed of a task at a given server depends on both the task and the server.

Our work is motivated by two important problems in radio communication networks. (1) Dynamic Spectrum Access. Transmitters are today able to exploit a large part of the radio spectrum, and can switch frequency bands rapidly. The service rate achieved on a link operating on a given frequency band depends on the load of this band (i.e., the number of competing links exploiting on the same band), and on the channel conditions which in turn depend on the band and the link (this phenomenon is known as *frequency selective fading*). When joining the system, transmitters are not aware of the load of each frequency band, and they select an arbitrary band (or may be that with the best channel conditions). However they may explore new bands, and migrate there if this improves their throughput. The overall system performance then strongly relies on the distributed resampling and switching strategy. (2) Access Point Selection. When users in a wireless network may attach to various access points, we get a similar situation. The throughput experienced by a user depends on the load of the selected access point, and on the geographical proximity of this access point. In both examples, the network can be modelled as an heterogenous parallel server system: servers correspond to either frequency bands or access points, and the heterogeneity stems from frequency selective fading or from the heterogenous channel conditions to the various access points.

In this paper, we consider a generic heterogenous system of parallel servers fairly sharing their resources in time among users (they implement a Processor Sharing discipline). Users initially pick a server randomly, and may later switch servers during their services. For such systems, we aim at identifying efficient and fair distributed resampling algorithms. We analyze the performance of the Random Local Search (RLS) algorithm introduced and studied in [1] in the specific case of homogenous systems (all users are served at the same speed at any server). Under RLS, a user resamples a server randomly selected at the instants of a Poisson process, and moves there if this improves her service rate. RLS is evaluated in both *closed* and *open* systems. In closed systems, the user population is fixed, and we compare the performance of RLS to that obtained under an ideal Proportionally Fair (PF) allocation of users to servers. We also study the rate at which RLS converges. In open systems, users arrive according to Poisson processes and leave upon service completion. In such scenarios, we are primarily interested in the throughput region of the RLS algorithm, i.e.,

*This work has been supported by the ERC grant 308267, VR, and SSF.

SIGMETRICS'15, June 15–19, 2015, Portland, OR, USA.

DOI: http://dx.doi.org/10.1145/2745844.2745861.

on the set of user arrival rates such that the system is stable under RLS. We compare this region to the maximal throughput region, defined as the set of arrival rates such that there exists a centralized load balancing scheme stabilizing the system. Using an usual terminology, a load resampling algorithm with maximal throughput region is called *throughput optimal*.

In [1], the performance of the RLS algorithm is investigated in the case of homogenous systems. The analysis of the latter systems is much simpler. Indeed in such systems, allocations where each server roughly handles the same number of users are efficient, and they are easy to identify and reach. In heterogenous systems, characterizing and finding efficient allocations turns out to be much more challenging, as users have individual preferences for servers (this problem ressembles a max-weight matching problem). Also note that in open homogenous systems, the design of throughput optimal algorithms is simplified by the fact that to stabilize the system, it is essentially sufficient to keep all servers active. In [1], this simple observation is used to prove that RLS is throughput optimal. A similar result in heterogenous systems would be more surprising, and much more difficult to establish.

Our contributions are as follows:

- Closed systems. Under RLS, the system dynamics correspond to those in an ordinal potential game where players (here users) sequentially and selfishly update their strategy (the selected server). Hence under RLS, the system converges to pure Nash Equilibriums (NEs). We study the efficiency of these NEs. We prove that when the user population grows large, pure NEs get closer to a Proportionally Fair (PF) allocation of users to servers. We provide an upper bound of the gap between the welfares of NEs and of this ideal allocation depending on user population. This gap scales at most as $\frac{S}{n}\log(n+S)$ where n and S denote the number of users and servers, respectively. We further study the time it takes, under RLS, for the system to reach the PF allocation within a certain margin. More precisely, when the user population is large enough, after a time $\frac{S\log(S)}{\epsilon}$, the expected system welfare under RLS is at a distance at most ϵ of the welfare of the PF allocation.

- Open systems. We establish that RLS is throughput optimal. To do so, we use the analysis of the system dynamics under RLS in closed systems to build an appropriate Lyapunov function. In this construction, we need to explicitly account for the interactions between user migrations across servers, and user population dynamics. This contrasts with the analysis of homogenous systems, where the throughput optimality of RLS was simply due to the fact that the algorithm keeps all servers active when the user population grows large.

- Our results in closed and open systems suggest that RLS is optimal when the system is heavily loaded. However, it might yield poor performance when the user population is small. To improve the performance in such scenarios, we propose mRLS, an extension of RLS where each user maintains a connection with $m > 1$ servers. All result are extended to this scenario. We finally consider a limiting regime where m grows large. In this regime, users behave as if they were bidding with a fixed budget to acquire services from the various servers. We devise an algorithm, referred to as Water-Filling (WF), under which users update their bids to the various servers in such a way that the resulting limiting allocation corresponds to the PF allocation. The WF algorithm is hence efficient even for small user populations, and we also establish its throughput optimality.

2. RELATED WORK

Distributed load resampling strategies can be analyzed using game theoretical techniques, and there have been many studies in the analysis of load balancing games, see [2] for a survey. Most of the analysis aim at characterizing the inefficiency of NEs in these games, through the notion of price of anarchy [3]. Researchers have also looked at the time it takes to balance the system under various strategies, including best-response or Nash dynamics (such strategies require that users are aware of the load at all servers) [4], and more distributed strategies, see [5] and references therein. Please also refer to [1] for a more detailed discussion. As far as we know, all existing results concern homogenous systems. In heterogenous systems, the underlying games are more involved, as they do not belong to the class of potential games [6–9]. This significantly complicates the analysis of the efficiency of NEs in the case of finite user population, and very little is known about the time it takes to reach these NEs. Our approach to study such games departs from existing techniques. The use of the ideal and pivotal PF allocation to analyze NEs is novel, and turns out to be powerful to derive results in both closed and open systems.

Accounting for the user population dynamics is rare in game theory. In [10], the authors analyze the performance of a logit algorithm in congestion games where the set of players may vary over time. However, these variations are exogenous, and the authors do not investigate the interaction between users' strategic behaviors and user population dynamics. Note however that handling this interaction is crucial to analyze the system stability. As already mentioned in the introduction, we extend the results of [1] to heterogenous systems. These extensions required the development of new methods. Of course, there is an abundant literature on the performance of classical load balancing schemes in open systems, where users are not allowed to switch servers. For example, in [11], the throughput optimality of the so called min drift routing policy is established. To our knowledge, except [1], natural load resampling strategies have not been studied in open systems. There is also an abundant literature on dynamic spectrum access and on the access point selection problem, see e.g. [12] and references therein. To our knowledge, none of this literature looks at the transient system behavior (convergence time), and at user population dynamics precisely.

3. MODELS

We consider a set $\mathcal{S}$ of S servers shared by n users. The latter are categorized into a set $\mathcal{K}$ of K classes. Users of class k may access to a subset $\mathcal{S}_k \subset \mathcal{S}$ of servers. We denote by $\mathcal{K}_s$ the set of classes whose users may be served by server s. At a given time t, the state of the system is represented by a vector $\boldsymbol{n}(t)$ describing the numbers of users of the different classes attached to the various servers. When the system state is $\boldsymbol{n} = (n_{ks})_{k\in\mathcal{K},s\in\mathcal{S}} \in \mathbb{N}^{K\times S}$, n_{ks} denotes the number of class-k users attached to server s. Note that $n_{ks} = 0$ if $s \notin \mathcal{S}_k$. For $s \in \mathcal{S}$, we further denote by $n^s = \sum_{k\in\mathcal{K}} n_{ks}$ the number of users attached to server s, and for $k \in \mathcal{K}$, $n_k = \sum_{s\in\mathcal{S}_k} n_{ks}$ is the number of class-k users.

An alternative way of representing the system state is to use the user population n and the proportions of users of different classes and attached to the various servers, $\boldsymbol{\alpha} = (\alpha_{ks})_{k\in\mathcal{K},s\in\mathcal{S}} \in \Delta^{K\times S}$, where Δ^p is the simplex of dimension p, i.e., $\boldsymbol{x} \in \Delta^p$ if $\sum_i x_i = 1$ and for all i, $x_i \geq 0$. We write $\boldsymbol{n} \sim (n, \boldsymbol{\alpha})$ if $n = \sum_k n_k$, and for all k, s, $n_{ks} = \alpha_{ks} n$.

Each server fairly shares its capacity in time among users attached to it. We consider heterogenous systems: users are served at different speeds at different servers. Specifically, let μ_{ks} be the service speed of a class-k user at server s. Hence, when the system is in state $\boldsymbol{n}$, each class-k user attached to server s is served at rate μ_{ks}/n^s. We denote by $\mu_{\min} = \min_{k\in\mathcal{K}} \min_{s\in\mathcal{S}_k} \mu_{ks} > 0$ and by $\mu_{\max} = \max_{k\in\mathcal{K}} \max_{s\in\mathcal{S}_k} \mu_{ks}$ the minimum and maximum speed at which users can be served, and define $\xi = \mu_{\max}/\mu_{\min}$.

3.1 Distributed Load Balancing Algorithms

Users have a myopic view of the system in the sense that they are aware of their current service rate, but they do not know the rate at which they would be served at other servers. We consider natural distributed load balancing strategies, where users independently resample and switch servers to selfishly improve their rates. The two first proposed strategies, referred to as Random Local Search (RLS) and Random Load Oblivious (RLO) algorithms, respectively, have been introduced in [1], and analyzed in homogenous scenarios where users are served at the same speed when attached to the same server, namely for all s and all $k \in \mathcal{K}_s$, μ_{ks} does not depend on k. We shall compare these two first algorithms to the Best Response (BR) algorithm, under which when a user decides to switch server, she picks the one offering the best service rate.

- *The RLS and RLO algorithms*. At the instants of a Poisson process of intensity β, a user randomly selects a new server (if the user is of class k, this choice is uniform over $\mathcal{S}_k$). Under RLS, she migrates to it if this would increase her service rate, whereas under RLO, she migrates to it blindly. Assume that the system is in state $\boldsymbol{n}$, and that a class-k user attached to server s considers switching servers. Let s' be the randomly selected server. Under RLS, the user migrates to s' if $\mu_{ks'}/(n^{s'}+1) > \mu_{ks}/n^s$, and under RLO, she migrates even if this would decrease her service rate.
- *The BR algorithm.* Each user switches servers at the instants of a Poisson process of intensity β. If the system is in state $\boldsymbol{n}$, when the opportunity arises, a class-k user attached to server s migrates to $s' \in \arg\max_{c\in\mathcal{S}_k} \mu_{kc}/(n^c + 1_{s'\neq s})$. Observe that to implement the BR algorithm, users need to know the service rates they would achieve at the various servers, which can be costly.

3.2 Closed Systems

In a *closed* system, the numbers of users of the various classes are fixed, and we are interested in the system dynamics under the RLS and BR algorithms (under the RLO algorithm, the loads at the various servers evolve somewhat arbitrarily, and this algorithm is not relevant here). Let $n_{[1-K]} = (n_1,\dots,n_K)$ represent the fixed population of users of the various classes, and let $\mathcal{N}(n_{[1-K]})$ denote the set of feasible system states having n_k class-k users for all $k \in \mathcal{K}$: $\mathcal{N}(n_{[1-K]}) = \{\boldsymbol{m} \in \mathbb{N}^{K\times S} : \forall k \in \mathcal{K}, \forall s \in \mathcal{S}, \sum_{s\in\mathcal{S}_k} m_{ks} = n_k, (m_{ks} > 0 \Rightarrow s \in \mathcal{S}_k)\}$.

To investigate the system state dynamics under the RLS and BR algorithms, we interpret $\boldsymbol{n}(t)$ as the set of strategies used at time t by the various players in a dynamic load balancing game. In this game, the set of pure strategies available to a class-k user or player is just the set of servers $\mathcal{S}_k$, and her payoff corresponds to her service rate. In what follows, we study the existence and efficiency of pure Nash Equilibriums in this load balancing game, and characterize the speed at which $\boldsymbol{n}(t)$ converges to these equilibriums under the RLS and BR algorithms.

3.3 Open Systems

In *open* systems, the user population evolves over time. Users of class k arrive in the system according to a Poisson process of intensity ρ_k, and each user has an exponentially distributed service requirement with unit mean. Upon arrival, a class-k user randomly selects a server uniformly at random in $\mathcal{S}_k$. User arrivals are hence characterized by a *load* vector $\boldsymbol{\rho} = (\rho_1,\dots,\rho_K)$. Users leave the system upon service completion, and in view of our assumptions, when the system is in state $\boldsymbol{n}$, a class-k user attached to server s leaves the system at rate μ_{ks}/n^s. Under the RLS, RLO, and BR algorithms, the system state $(\boldsymbol{n}(t))_{t\geq 0}$ is a continuous-time Markov chain, and we are interested in providing conditions under which this process is positive recurrent. More precisely, we analyzed the throughput region of each of the three algorithms, defined as the set of load vectors $\boldsymbol{\rho}$ such that $(\boldsymbol{n}(t))_{t\geq 0}$ is positive recurrent.

Define the maximal throughput region as the set of $\boldsymbol{\rho}$ such that there exists a centralized load balancing algorithm (possibly depending on $\boldsymbol{\rho}$) under which $(\boldsymbol{n}(t))_{t\geq 0}$ is positive recurrent. The following lemma is classical (see e.g. [13]):

LEMMA 3.1. *$\boldsymbol{\rho}$ lies in the maximal throughput region if and only if there exists $\boldsymbol{\lambda} \in \mathcal{R}$ such that $\boldsymbol{\rho} < \boldsymbol{\lambda}$ component-wise, where*

$$\mathcal{R} = \{\boldsymbol{r} \in \mathbb{R}_+^K : \exists \boldsymbol{\omega} \in \Omega : \forall k \in \mathcal{K}, r_k = \sum_{s\in\mathcal{S}_k} \omega_{ks}\mu_{ks}\},$$

and where Ω is the set of $\boldsymbol{\omega} = (\omega_{ks})_{k\in\mathcal{K},s\in\mathcal{S}}$ such that (i) for all k, s, $\omega_{ks} \geq 0$, (ii) for all s, $\omega_{ks} = 0$ if $k \notin \mathcal{K}_s$, and (iii) $\sum_{k\in\mathcal{K}_s} \omega_{ks} \leq 1$.

In the above lemma, ω_{ks} may be interpreted as the proportion of time s serves class-k users. It is worth remarking that $\mathcal{R}$ is also the maximal throughput region when jointly considering user association (or load balancing) strategy and service discipline policies at each server. We say that a load resampling algorithm is throughput optimal if its throughput region coincides with $\mathcal{R}$.

4. CLOSED SYSTEMS

This section is devoted to the analysis of the system state dynamics under the RLS and BR algorithms in closed systems. The numbers of users of various classes are fixed and represented by a vector $n_{[1-K]} = (n_1,\dots,n_K)$ where n_k is the number of class-k users. For all $t \geq 0$, $\boldsymbol{n}(t) \in \mathcal{N}(n_{[1-K]})$. The initial allocation of users to servers $\boldsymbol{n}(0)$ is arbitrary. As mentioned previously, we may interpret the evolution of $\boldsymbol{n}(t)$ under RLS and BR as the evolving strategies of players or users in a dynamic load balancing game. In this game, $\boldsymbol{n} \in \mathcal{N}(n_{[1-K]})$ constitutes a pure NE if and only if:

$$\forall k \in \mathcal{K}, \forall s \in \mathcal{S}_k : n_{ks} > 0, \forall s' \in \mathcal{S}_k, \frac{\mu_{ks}}{n^s} \geq \frac{\mu_{ks'}}{n^{s'}+1}.$$

The above inequality states that in state $\boldsymbol{n}$, a class-k user attached to server s has no incentive to move to server s'. We show that pure NEs exist, and that under both RLS and BR, $\boldsymbol{n}(t)$ converges to a pure NE as t grows large. The existence of pure NEs is simply due to the fact that the game admits an *ordinal potential* function. We also study the efficiency of the pure NEs, and the rate at which $\boldsymbol{n}(t)$ converges to an equilibrium under the RLS and BR algorithms. We establish that for large user populations, the equilibriums are close to the proportionally fair allocation, and provide an upper bound of the distance to this ideal allocation depending on the numbers of users and servers.

The analysis of the rate of convergence to equilibrium under RLS and BR turns out to be challenging. This is mainly due to the system heterogeneity. Recall that in [1], this rate of convergence was studied for homogenous systems with servers of equal capacities, i.e., $\mu_{ks} = 1$ for all k and s, which significantly simplifies the analysis. To analyze RLS and BR in heterogenous systems, our idea is to track the evolution over time of the distance between $\boldsymbol{n}(t)$ and the ideal proportionally fair allocation. To this aim, we establish a relationship between the expected drifts in social welfare (defined with logarithmic utilities) and in the ordinal potential. In turn, this allows us to quantify the rate of convergence of the system state under the RLS and BR algorithms when the user population is typically large.

4.1 Ordinal Potential and Social Welfare

4.1.1 An ordinal potential game

In our load balancing game, the system state $\boldsymbol{n}$ determines the strategies played by all users (n_{ks} is the number of class-k users playing or selecting server s). As shown in [14], due to the system heterogeneity, the game does not belong to the class of potential games [15]. Indeed in general, one cannot find a set of increasing functions $u_k : \mathbb{R}_+ \to \mathbb{R}$, and a potential function $\psi : \mathcal{N}(n_{[1-K]}) \to \mathbb{R}$ such that: for any $k \in \mathcal{K}$ and any $s, s' \in \mathcal{S}_k$, $\psi(\boldsymbol{n} + e_{ks'} - e_{ks}) - \psi(\boldsymbol{n}) = u_k(\frac{\mu_{ks'}}{n^{s'}+1}) - u_k(\frac{\mu_{ks}}{n^s})$ for all $\boldsymbol{n} \in \mathcal{N}(n_{[1-K]})$. Here $\boldsymbol{n} + e_{ks'} - e_{ks}$ denotes the system state obtained from $\boldsymbol{n}$ by moving one class-k user from server s to server s'. In fact, our load balancing game is a *congestion game with player-specific utilities* as introduced in [6]. It can be readily checked that there exists a function $\psi : \mathcal{N}(n_{[1-K]}) \to \mathbb{R}$ such that for all $\boldsymbol{n} \in \mathcal{N}(n_{[1-K]})$, for all k, s, s' such that $n_{ks} > 0$, $\frac{\mu_{ks}}{n^s} < \frac{\mu_{ks'}}{n^{s'}+1} \Rightarrow \psi(\boldsymbol{n}) < \psi(\boldsymbol{n} + e_{ks'} - e_{ks})$. Such a function ψ is referred to an ordinal potential function. A possible ordinal potential function ψ is defined by: for all $\boldsymbol{n} \in \mathcal{N}(n_{[1-K]})$,

$$\psi(\boldsymbol{n}) = -\sum_{s\in\mathcal{S}} \log(n^s!) + \sum_{s\in\mathcal{S}}\sum_{k\in\mathcal{K}} n_{ks}\log(\mu_{ks}).$$

The existence of an ordinal potential function ensures that pure NEs exists, and that the system dynamics under the RLS and BR algorithms converge to a pure NE [16].

4.1.2 Proportionally Fair allocation

When the user population is large, $\boldsymbol{n}$ is a pure NE if and only if: for all k, let $s \in \mathcal{S}_k$ such that $n_{ks} > 0$, then for any $s' \in \mathcal{S}_k$, $n_{ks'} > 0$ iff $\frac{\mu_{ks'}}{n^{s'}} = \frac{\mu_{ks}}{n^s}$, and $n_{ks'} = 0$ iff $\frac{\mu_{ks'}}{n^{s'}} < \frac{\mu_{ks}}{n^s}$. These conditions actually coincide with the KKT conditions of the following convex program:

$$\begin{aligned} \text{maximize} \quad & W(\boldsymbol{x}) = \sum_{k\in\mathcal{K}}\sum_{s\in\mathcal{S}_k} x_{ks}\log\left(\frac{\mu_{ks}}{\sum_{k\in\mathcal{K}} x_{ks}}\right) \\ \text{over} \quad & \boldsymbol{x} \in \mathcal{N}^R(n_{[1-K]}), \end{aligned} \tag{1}$$

where $\mathcal{N}^R(n_{[1-K]}) = \{\boldsymbol{y} \in \mathbb{R}_+^{K\times S} : \forall k, s, \sum_{s\in\mathcal{S}_k} y_{ks} = n_k,$ $(y_{ks} > 0 \Rightarrow s \in \mathcal{S}_k)\}$. In what follows, we denote by $x^\star$ the solution of (1), and $W^\star = W(x^\star)$. The above observation suggests that when the number of users grows large, the pure NEs become efficient in the sense that the allocation of users to servers is proportionally fair [17]. In the following subsection, we formalize this observation more precisely.

Define the social welfare in state $\boldsymbol{n} \in \mathcal{N}(n_{[1-K]})$ as:

$$W(\boldsymbol{n}) = \sum_{k\in\mathcal{K}}\sum_{s\in\mathcal{S}_k} n_{ks}\log\left(\frac{\mu_{ks}}{n^s}\right).$$

The proportionally fair allocation $\boldsymbol{n}^\star$ maximizes $W(\boldsymbol{n})$ over all $\boldsymbol{n} \in \mathcal{N}(n_{[1-K]})$. Note that typically, the social welfare scales as $-n\log(n)$ when n is large (the service of a user scales as $1/n$). Hence it may be more informative to work on a scaled version of the welfare. For $\boldsymbol{\alpha} \in \Delta^{K\times S}$, define $w(\boldsymbol{\alpha})$ as:

$$w(\boldsymbol{\alpha}) = \sum_{k\in\mathcal{K}}\sum_{s\in\mathcal{S}_k} \alpha_{ks}\log\left(\frac{\mu_{ks}}{\sum_\ell \alpha_{\ell s}}\right).$$

Note that when $\boldsymbol{n} \sim (n, \boldsymbol{\alpha})$, we have: $W(\boldsymbol{n}) = nw(\boldsymbol{\alpha}) - n\log(n)$. Now observe that $x^\star = (n, \boldsymbol{\alpha}^\star)$ if and only if $\boldsymbol{\alpha}^\star$ solves the following convex program:

$$\begin{aligned} \text{maximize} \quad & w(\boldsymbol{\alpha}) = \sum_{k\in\mathcal{K}}\sum_{s\in\mathcal{S}_k} \alpha_{ks}\log\left(\frac{\mu_{ks}}{\sum_\ell \alpha_{\ell s}}\right) \\ \text{subject to} \quad & \forall k, \sum_{s\in\mathcal{S}_k} \alpha_{ks} = \frac{n_k}{n}, \quad \forall s, \alpha_{ks} \geq 0. \end{aligned} \tag{2}$$

$\alpha^\star$ represents the proportionally fair allocation of users to servers when the system size n is large.

It will also turn useful to introduce a slightly different notion of social welfare: $V(\boldsymbol{n}) = \sum_{k\in\mathcal{K}} n_k \log\left(\sum_{s\in\mathcal{S}_k} \frac{n_{ks}}{n_k} \times \frac{\mu_{ks}}{n^s}\right)$. $V(\boldsymbol{n})$ may be interpreted as the social welfare of a system where users of a given class fairly share the sum of the services rates of users of the same class. The following lemma relates the welfares $W(\boldsymbol{n})$ and $V(\boldsymbol{n})$ and states that these welfares coincide when they are maximized.

LEMMA 4.1. *(i) For all $\boldsymbol{n}$, $V(\boldsymbol{n}) \geq W(\boldsymbol{n})$.*
(ii) Let $\boldsymbol{x}' \in \arg\max_{x\in\mathcal{N}^R(n_{[1-K]})} V(\boldsymbol{x})$, and $V^\star = V(\boldsymbol{x}')$, then $W^\star = W(x') = V^\star$, i.e., $x^\star = x'$.

4.2 Efficiency of Nash Equilibriums

Pure NEs can be quite inefficient when the user population is small. For example, consider a system with two users and two servers, where users have a different preferred server (say user i has a greater service speed at server i, for $i = 1, 2$). It might well be that the allocation where both users are attached to their un-preferred server is a pure NE, which is indeed inefficient.

When the user population grows large, the allocations corresponding to pure NEs become more efficient, and get close to the ideal proportionally fair allocation. Our objective here is to quantify this observation precisely: we provide upper bounds on the difference between the social welfare achieved under the proportionally fair allocation and under pure NEs.

4.2.1 Average potential drift

We first establish a crucial result relating the average drift in the ordinal potential function ψ to the system welfare under the RLS algorithm. Let $(t_i)_{i\geq 1}$ be the increasing sequence of epochs at which one user has the opportunity to switch servers under the RLS algorithm. This sequence is the superposition of n Poisson processes of intensity β, and hence corresponds to the instants of a Poisson process of intensity $n\beta$.

THEOREM 4.1. *For all $i \geq 1$, the expected drift in the potential under RLS after the update taking place at time t_{i+1} satisfies:*

$$\begin{aligned} &\mathbb{E}\big[\psi(\boldsymbol{n}(t_{i+1})) - \psi(\boldsymbol{n}(t_i)) | \boldsymbol{n}(t_i)\big] \\ &\qquad \geq \frac{1}{nS}\Big(W^\star - W(\boldsymbol{n}(t_i)) - S\log\big(e\xi(n+S)\big)\Big). \end{aligned} \tag{3}$$

4.2.2 Social Welfare in pure NEs

Assume that at time t_i, the system has reached a pure NE. Then the subsequent average potential drifts vanish, i.e.,

$$\mathbb{E}\left[\psi(\boldsymbol{n}(t_{i+1})) - \psi(\boldsymbol{n}(t_i)) | \boldsymbol{n}(t_i)\right] = 0.$$

As a consequence, from the previous theorem, we deduce that:

COROLLARY 4.1. *If $\boldsymbol{n}$ is a pure NE, then*

$$W^\star - W(\boldsymbol{n}) \le S\log\big(e\xi(n+S)\big), \tag{4}$$

$$V^\star - V(\boldsymbol{n}) \le S\log\big(e\xi(n+S)\big). \tag{5}$$

(5) is obtained by combining (4) and the claim (i) of Lemma 4.1. To interpret Corollary 4.1, it is convenient to look at the scaled version of the social welfare. Let $\boldsymbol{n} \sim (n, \boldsymbol{\alpha})$ be a pure NE, then according to (4), we have: $w(\boldsymbol{\alpha}^\star) - w(\boldsymbol{\alpha}) \le \frac{S}{n}\log\big(e\xi(n+S)\big)$. The above inequality implies that when the user population grows large, the allocation $\boldsymbol{\alpha}$ of users to servers in a pure NE converges to the proportionally fair allocation $\boldsymbol{\alpha}^\star$. The inequality also quantifies how fast this convergence occurs.

4.3 Convergence Rate

We are now interested in the system dynamics under the RLS and BR algorithms, and in particular, we wish to analyze the rate at which the system approaches pure NEs under these algorithms. To do so, we take a detour, and provide an upper bound of the difference between $W^\star$ and the welfare of the system at time t under one of these algorithms. In turn, in view of Corollary 4.1, this allows us to estimate the rate of convergence of RLS and BR when the user population is large.

THEOREM 4.2. *For any initial condition $\boldsymbol{n}(0)$, under the RLS algorithm, we have for all t:*

$$\mathbb{E}\left[W^\star - W(\boldsymbol{n}(t))\right] \le \frac{nS}{\beta t}\log(S\xi) + S\log\big(e\xi n(n+S)\big).$$

Let $\boldsymbol{\alpha}(t)$ denote the fractions of users of various classes attached to the different servers at time t under the RLS algorithm, i.e., $\boldsymbol{n}(t) \sim (n, \boldsymbol{\alpha}(t))$. Theorem 4.2 states that:

$$\mathbb{E}\left[w(\boldsymbol{\alpha}^\star) - w(\boldsymbol{\alpha}(t))\right] \le \frac{S}{\beta t}\log(S\xi) + \frac{S}{n}\log\big(e\xi n(n+S)\big).$$

Applying Markov inequality, we deduce that for large systems ($n \to \infty$), after time t, the system state has a scaled welfare within ϵ of $w(\boldsymbol{\alpha}^\star)$ with probability at least $1-\delta$ as soon as $t \ge \frac{S\log(\xi S)}{\epsilon\delta}$. In particular, the convergence time towards the proportionally fair allocation does not depend on the user population.

Under the BR algorithm, the system converges typically more rapidly than under RLS, as stated in the following theorem (whose proof is omitted – it is similar to that of Theorem 4.2).

THEOREM 4.3. *For any initial condition $\boldsymbol{n}(0)$, under the BR algorithm, we have for all t:*

$$\mathbb{E}\left[W^\star - W(\boldsymbol{n}(t))\right] \le \frac{n}{\beta t}\log(S\xi) + S\log\big(e\xi n(n+S)\big).$$

4.4 Proofs

4.4.1 Proof of Theorem 4.1

Under the RLS algorithm, at time t_i, the probability that a class-k user attached to server s has the opportunity to switch servers is $n_{ks}(t_i)/n$. The probability that the probed server is $c \in \mathcal{S}_k$ is $1/S_k \ge 1/S$ where S_k is the cardinality of $\mathcal{S}_k$. This user migrates to the new server only if $\frac{\mu_{kc}}{n^c(t_i)+1} > \frac{\mu_{ks}}{n^s(t_i)}$. Hence the expected drift in the ordinal potential satisfies: for any $\boldsymbol{n} \in \mathcal{N}(n_{[1-K]})$,

$$\begin{aligned}
&\mathbb{E}\big[\psi(\boldsymbol{n}(t_{i+1})) - \psi(\boldsymbol{n}(t_i)) | \boldsymbol{n}(t_i) = \boldsymbol{n}\big] \\
&\quad\ge \sum_{k\in\mathcal{K}}\sum_{s\in\mathcal{S}_k}\sum_{c\in\mathcal{S}_k}\frac{n_{ks}}{nS}\max\{0, \log\frac{\mu_{kc}}{n^c+1} - \log\frac{\mu_{ks}}{n^s}\} \\
&\quad\ge \sum_{k\in\mathcal{K}}\sum_{s\in\mathcal{S}_k}\frac{n_{ks}}{nS}\max\{0, \max_{c\in\mathcal{S}_k}\log\frac{\mu_{kc}}{n^c+1} - \log\frac{\mu_{ks}}{n^s}\} \\
&\quad\ge \sum_{k\in\mathcal{K}}\left(\frac{n_k}{nS}\max_{s\in\mathcal{S}_k}\log\frac{\mu_{ks}}{n^s+1} - \sum_{s\in\mathcal{S}_k}\frac{n_{ks}}{nS}\log\frac{\mu_{ks}}{n^s}\right).
\end{aligned}$$

It remains to show that:

$$\begin{aligned}
&\sum_{k\in\mathcal{K}}\left(n_k\max_{s\in\mathcal{S}_k}\log\frac{\mu_{ks}}{n^s+1} - \sum_{s\in\mathcal{S}_k}n_{ks}\log\frac{\mu_{ks}}{n^s}\right) \\
&\qquad\qquad \ge W^\star - W(\boldsymbol{n}(t_i)) - S\log\big(e\xi(n+S)\big).
\end{aligned} \tag{6}$$

Let $\boldsymbol{n} \in \mathcal{N}(n_{[1-K]})$, and let $\boldsymbol{n}'$ be a system state such that each server has one more users than in state $\boldsymbol{n}$. Hence $n' = n + S$, and for all s, $n'^s = n^s + 1$. For all k and s, $n_{ks} \le n'_{ks} \le n_{ks}+1$. Define $\boldsymbol{\delta} = (\delta_{ks} = n'_{ks} - n_{ks})_{k\in\mathcal{K}, s\in\mathcal{S}}$. Let $\mathcal{N}(n'_{[1-K]})$ the set of all possible states of a closed system starting from $\boldsymbol{n}'$. Define $\boldsymbol{x}'^\star = \arg\max_{\boldsymbol{x}\in\mathcal{N}^R(n'_{[1-K]})} W(\boldsymbol{x})$. We have:

$$W^\star - W(\boldsymbol{n}) = A + B + W(\boldsymbol{n}') - W(\boldsymbol{n}), \tag{7}$$

where $A = W^\star - W(\boldsymbol{x}'^\star)$ and $B = W(\boldsymbol{x}'^\star) - W(\boldsymbol{n}')$. From the definitions of $W^\star$ and $W(\boldsymbol{x}'^\star)$,

$$\begin{aligned}
A &= W^\star - \max_{\boldsymbol{x}'\in\mathcal{N}^R(n'[1-K])} W(\boldsymbol{x}') \\
&\le W^\star - W(\boldsymbol{x}^\star + \boldsymbol{\delta}) \;\le\; \nabla W(\boldsymbol{x}^\star + \boldsymbol{\delta})^T \cdot (-\boldsymbol{\delta}) \\
&\le -S\left(\log\frac{\mu_{\min}}{n+S} - 1\right),
\end{aligned} \tag{8}$$

where we used the concavity of W in the second inequality. Again, using the concavity of W, B satisfies:

$$\begin{aligned}
B &\le \nabla W(\boldsymbol{n}')^T \cdot (\boldsymbol{x}'^\star - \boldsymbol{n}') \\
&\le \max_{\boldsymbol{x}'\in\mathcal{N}^R(n'[1-K])} \nabla W(\boldsymbol{n}')^T \cdot \boldsymbol{x}' - \nabla W(\boldsymbol{n}')^T \cdot \boldsymbol{n}' \\
&\le \sum_{k\in\mathcal{K}} n'_k \max_{s\in\mathcal{S}_k} \log\frac{\mu_{ks}}{n^s+1} - \sum_{k\in\mathcal{K}}\sum_{s\in\mathcal{S}_k} n'_{ks}\log\frac{\mu_{ks}}{n^s+1} \\
&\le \sum_{k\in\mathcal{K}} n_k \max_{s\in\mathcal{S}_k}\log\frac{\mu_{ks}}{n^s+1} + S\log\mu_{\max} - W(\boldsymbol{n}')
\end{aligned} \tag{9}$$

Combining (7), (8), and (9), we get (6). □

4.4.2 Proof of Theorem 4.2

To prove the theorem, we first state and prove three preliminary lemmas. The first lemma provides an upper bound of the difference between the maximum and the minimal values of the ordinal potential function ψ over $\mathcal{N}(n_{[1-K]})$.

LEMMA 4.2. *Let $\psi_{\max} = \max_{\boldsymbol{n}\in\mathcal{N}(n_{[1-K]})}\psi(\boldsymbol{n})$ and $\psi_{\min} = \min_{\boldsymbol{n}\in\mathcal{N}(n_{[1-K]})}\psi(\boldsymbol{n})$. Then:*

$$\psi_{\max} - \psi_{\min} \le n\log(S\xi).$$

Proof. We first provide a lower and an upper bound of $\sum_s \log(n^s!)$. To this aim, we note that $n! = \Gamma(n+1)$ and use the convexity of $\log\Gamma(\,\cdot)$. We have:

$$\begin{aligned}\sum_{s\in\mathcal{S}}\log(n^s!) &= \sum_{s\in\mathcal{S}}\log(n^s!) + n\log S - n\log S\\ &= \sum_{s\in\mathcal{S}}\sum_{m=1}^{n^s}\log(Sm) - n\log S\\ &\overset{(a)}{\geq} \frac{1}{S}\sum_{s\in\mathcal{S}}\log((Sn^s)!) - n\log S\\ &\overset{(b)}{\geq} \frac{1}{S}\sum_{s\in\mathcal{S}}\log((Sn^s)!) - n\log S\\ &\quad - \frac{1}{S}\sum_{s\in\mathcal{S}}\log\Gamma(\,Sn^s+1) + \log(n!)\\ &= -\,n\log S + \log(n!),\end{aligned}$$

where (a) comes the combinatorial inequality $\frac{(Sn)!}{(n!)^S} \leq S^{Sn}$, and (b) is due to the convexity of $\log\Gamma(\,\cdot)$.

The upper bound of $\sum_{s\in\mathcal{S}}\log(n^s!)$ follows from the fact that: $\sum_{s\in\mathcal{S}}\log(n^s!) \leq \log(n!)$. For $\boldsymbol{n}\in\mathcal{N}(n_{[1-K]})$, $\psi(\boldsymbol{n})$ satisfies

$$\begin{aligned}\psi(\boldsymbol{n}) &= -\sum_{s\in\mathcal{S}}\log(n^s!) + \sum_{k\in\mathcal{K}}\sum_{s\in\mathcal{S}_k} n_{ks}\log(\mu_{ks})\\ &\leq n\log S - \log(n!) + \sum_{k\in\mathcal{K}} n_k \max_{s\in\mathcal{S}_k}\log(\mu_{ks}),\end{aligned}$$

and

$$\psi(\boldsymbol{n}) \geq -\log(n!) + \sum_{k\in\mathcal{K}} n_k \min_{s\in\mathcal{S}_k}\log(\mu_{ks}).$$

We deduce that:

$$\psi_{\max} - \psi_{\min} \leq n\log S + \sum_{k\in\mathcal{K}} n_k \log\frac{\max_{s\in\mathcal{S}_k}\mu_{ks}}{\min_{s\in\mathcal{S}_k}\mu_{ks}},$$

which concludes the proof. □

In the next lemma, we provide an upper bound of the minimum average gap between the maximum welfare $W^\star$ and the welfare after $i \leq I$ updates under the RLS algorithm.

LEMMA 4.3. *Let $I\in\mathbb{N}$. We have:*

$$\begin{aligned}&\min_{i\leq I}\mathbb{E}\left[W^\star - W(\boldsymbol{n}(t_i))\right]\\ &\quad\leq \frac{nS}{I+1}\cdot n\log(S\xi) + S\log\big(e\xi(n+S)\big).\end{aligned}$$

Proof. From Theorem 4.1, we have:

$$\begin{aligned}&\mathbb{E}\left[\psi(\boldsymbol{n}(t_{I+1})) - \psi(\boldsymbol{n}(0))\right]\\ &= \sum_{i=0}^{I}\mathbb{E}\left[\psi(\boldsymbol{n}(t_{i+1})) - \psi(\boldsymbol{n}(t_i))\right]\\ &\geq \sum_{i=0}^{I}\mathbb{E}\left[\frac{1}{nS}\Big(W^\star - W(\boldsymbol{n}(t_i)) - S\log\big(e\xi(n+S)\big)\Big)\right].\end{aligned}$$

By Lemma 4.2, $\psi(\boldsymbol{n}(t_{I+1})) - \psi(\boldsymbol{n}(0)) \leq n\log(S\xi)$. Thus,

$$\begin{aligned}&\min_{i\leq I}\mathbb{E}[W^\star - W(\boldsymbol{n}(t_i))]\\ &\leq \frac{nS}{I+1}\mathbb{E}[\psi(\boldsymbol{n}(t_{I+1})) - \psi(\boldsymbol{n}(0))] + S\log(e\xi(n+S))\\ &\leq \frac{nS}{I+1}n\log(S\xi) + S\log(e\xi(n+S)).\end{aligned}$$

□

Our third lemma provides an upper bound of the evolution over time of the difference between the social welfare and the potential.

LEMMA 4.4. *For all i and j,*

$$\big|\big(W(\boldsymbol{n}(t_i)) - \psi(\boldsymbol{n}(t_i))\big) - \big(W(\boldsymbol{n}(t_j)) - \psi(\boldsymbol{n}(t_j))\big)\big| \leq S\log n.$$

Proof. By definition,

$$W(\boldsymbol{n}) - \psi(\boldsymbol{n}) = \sum_{s\in\mathcal{S}}\Big(\log(n^s!) - n^s\log(n^s)\Big).$$

Using $\int_1^{n^s}\log(x)dx \leq \log(n^s!) \leq \int_1^{n^s+1}\log(x)dx$, we get:

$$\begin{aligned}&W(\boldsymbol{n}) - \psi(\boldsymbol{n})\\ &\leq \sum_{s\in\mathcal{S}}\Big((n^s+1)\log(n^s+1) - (n^s+1) - n^s\log(n^s)\Big)\\ &= \sum_{s\in\mathcal{S}} n^s\log(1+\frac{1}{n^s}) + \sum_{s\in\mathcal{S}}\log(1+n^s) - n - S\\ &\leq S + S\log n - n,\end{aligned}$$

and

$$\begin{aligned}W(\boldsymbol{n}) - \psi(\boldsymbol{n}) &\geq \sum_{s\in\mathcal{S}}(n^s\log n^s - n^s + 1 - n^s\log n^s)\\ &\geq S - n.\end{aligned}$$

We conclude that for all i and j,

$$\big|\big(W(\boldsymbol{n}(t_i)) - \psi(\boldsymbol{n}(t_i))\big) - \big(W(\boldsymbol{n}(t_j)) - \psi(\boldsymbol{n}(t_j))\big)\big| \leq S\log n.$$

□

We are now ready to prove Theorem 4.2. Let τ be the number of times users get an opportunity to switch servers up to time t under the RLS algorithm. From Lemma 4.3, given $\tau = I$, there exists $i^\star \leq I$ such that:

$$\begin{aligned}&\mathbb{E}\left[W^\star - W(\boldsymbol{n}(t_{i^\star}))|\tau = I\right]\\ &\leq \frac{nS}{I+1}\cdot n\log(S\xi) + S\log(e\xi(n+S)),\end{aligned}$$

and, by Lemma 4.4, for all $i^\star \leq i \leq I$,

$$W(\boldsymbol{n}(t_i)) \geq W(\boldsymbol{n}(t_{i^\star})) - \psi(\boldsymbol{n}(t_{i^\star})) + \psi(\boldsymbol{n}(t_i)) - S\log n.$$

Thus,

$$\begin{aligned}&\mathbb{E}\left[W^\star - W(\boldsymbol{n}(t))|\tau = I\right]\\ &\leq \mathbb{E}\left[W^\star - W(\boldsymbol{n}(t_{i^*}))\right]\\ &\qquad + \mathbb{E}\left[\psi(\boldsymbol{n}(t_{i^*})) - \psi(\boldsymbol{n}(t_i))\right] + S\log(n)\\ &\overset{(a)}{\leq} \mathbb{E}\left[W^\star - W(\boldsymbol{n}(t_{i^*}))\right] + S\log(n)\\ &\leq \frac{n^2S}{I+1}\log(S\xi) + S\log(e\xi(n+S)) + S\log(n),\end{aligned}$$

where (a) comes from the fact that the potential increases over time. Now, since τ has a Poisson distribution with mean $\lambda = \beta nt$, we have $\mathbb{E}[\frac{1}{I+1}] = \sum_{I=0}^{\infty}\frac{\lambda^I e^{-\lambda}}{I!(I+1)} \leq \frac{1}{\lambda}$.

Finally, we obtain:

$$\mathbb{E}\left[W^\star - W(\boldsymbol{n}(t))\right] \leq \frac{Sn}{\beta t}\log(S\xi) + S\log(e\xi n(n+S)).$$

□

5. OPEN SYSTEMS

This section is devoted to open systems: users arrive in the system randomly and leave upon service completion. Refer to §3.3 for a full description of the model.

5.1 Throughput Regions

In the following two theorems, we characterize the throughput region (i.e., the set of arrivals rates stabilizing the system) of the RLO, RLS, and BR algorithms. Define:

$$\mathcal{R}^{\text{RLO}} = \{\boldsymbol{r} \in \mathbb{R}_+^K : \exists \boldsymbol{\theta} \in \Delta^K : \forall k \in \mathcal{K}, \quad r_k \le \sum_{s \in \mathcal{S}_k} \mu_{ks} \frac{\theta_k / S_k}{\sum_{\ell \in \mathcal{K}_s} \theta_\ell / S_\ell} \},$$

THEOREM 5.1. *(Throughput region of RLO) Assume that there exists $\boldsymbol{r} \in \mathcal{R}^{\text{RLO}}$ such that $\boldsymbol{\rho} < \boldsymbol{r}$, then under the RLO algorithm, the Markov process $(\boldsymbol{n}(t))_{t\ge 0}$ is positive Harris recurrent. Conversely, if $\boldsymbol{\rho} > \boldsymbol{r}$ for all $\boldsymbol{r} \in \mathcal{R}^{\text{RLO}}$, the process is transient.*

THEOREM 5.2. *(Throughput Optimality of RLS and BR) Assume that there exists $\boldsymbol{r} \in \mathcal{R}$ such that $\boldsymbol{\rho} < \boldsymbol{r}$, then under the RLS (or BR) algorithm, the Markov process $(\boldsymbol{n}(t))_{t\ge 0}$ is positive Harris recurrent.*

In view of Lemma 3.1 and Theorem 5.2, RLS is throughput optimal. Hence a simple fully distributed and selfish load balancing scheme is able to stabilize the system whenever this is at all possible. This seems surprising. In closed systems, RLS converges to pure NEs, and the latter become efficient only when the user population is large. In addition, even if these pure NEs are efficient in heavily loaded systems, it takes time to reach efficient allocations when the user population changes (there is a priori no time-scale separation). The proof of Theorem 5.2 exploits the results derived in the previous section about the efficiency of pure NEs and the convergence rate to these NEs under the RLS algorithm in closed systems. These results allow us to build a Lyapunov function whose drift becomes negative when the user population is large.

In homogenous systems, i.e., when each server offers the same service speed to all users (μ_{ks} does not depend on k), RLO and RLS are throughput optimal [1]. The throughput optimality of RLS directly follows from the observation that to stabilize the system, it is enough to keep all servers active. In particular, the stability analysis of RLS in homogenous systems does not require a detailed analysis of closed systems. This contrasts with the case of heterogenous systems, where characterizing the throughput region of RLS calls for a more precise understanding of the way user migrations and user arrivals and departures interact.

In heterogenous systems, the RLO algorithm is no longer throughput optimal as stated in Theorem 5.1. This intuitive result was first mentioned in [13]. To prove it, we can combine the coupling argument used in [1] (to analyze the stability of RLO in homogenous systems) and the Lyapunov function used in [13]. The proof is omitted here. Note that to analyze the stability of systems with user migrations (whose rate is proportional to the number of users), it is wise not to consider fluid limits, as the corresponding limit theorems are extremely challenging to justify [18], see [1] for a detailed discussion. The inefficiency of the RLO algorithm can be illustrated through the example of a simple symmetric system with two user classes and two servers: $\mu_{11} = \mu_{22}$, $\mu_{12} = \mu_{21} < \mu_{11}$, and $\rho_1 = \rho_2 = \rho$. In this case, it can be readily shown that $(\rho,\rho) \in \mathcal{R}^{\text{RLO}}$ if and only if $\rho \le \frac{1}{2}(\mu_{11}+\mu_{12})$, whereas $(\rho,\rho) \in \mathcal{R}$ as long as $\rho \le \mu_{11}$. Hence when $\mu_{12} \ll \mu_{11}$, the system can support almost twice more traffic under RLS than under RLO.

5.2 Proofs

5.2.1 Stability of Markov processes

To investigate stability, we use a Foster-Lyapunov criterion for positive Harris recurrence of Markov processes with values in a locally compact and separable metric space $(\mathcal{X}, \mathcal{B}(\mathcal{X}))$ – refer to [19] for more details. Here our process has a countable state space, so we do not really need to apply this general criterion; however, we use it in the proofs of results in Section 7. Let $\Phi = (\Phi(t))_{t\ge 0}$ be a time-homogeneous Markov process with state space $(\mathcal{X}, \mathcal{B}(\mathcal{X}))$ and transition functions P^t. Assume that Φ is non-explosive (which is always the case in our systems because the total arrival rate is bounded). Let $\mathcal{A}$ be the generator of Φ, i.e., for all real-valued measurable function F on $\mathcal{X}$,

$$\mathcal{A}F(x) := \lim_{h\downarrow 0} \frac{\mathbb{E}\left[F(\Phi(h)) - F(\Phi(0)) | \Phi(0) = x\right]}{h}.$$

Further define, for a given distribution ν on $\mathbb{R}_+$, the Markov transition function K_ν as: $K_\nu(x,\cdot) := \int P^t(x,\cdot)\nu(dt)$. A set $C \in \mathcal{B}(\mathcal{X})$ is called *ϕ_ν-petite* if ϕ_ν is a non-trivial measure on $\mathcal{B}(\mathcal{X})$ and ν is a distribution on $(0,\infty)$ which satisfy $K_\nu(x,\cdot) \ge \phi_\nu(\cdot)$ for all $x \in C$.

For all $A \in \mathcal{B}(\mathcal{X})$, define the stopping time $\tau_A = \inf\{t \ge 0 : \Phi(t) \in A\}$. Φ is *Harris recurrent* if there exists some σ-finite measure ν such that $P\{\tau_A < \infty | \Phi(0) = x\} = 1$ for any $x \in \mathcal{X}$, whenever $\nu(A) > 0$. If $\boldsymbol{\Phi}$ is Harris recurrent, it has a unique invariant measure π. If, in addition, π is finite, $\boldsymbol{\Phi}$ is called *positive Harris recurrent*.

THEOREM 5.3 (THEOREM 4.2 [19]). *Let Φ be a non-explosive Markov process. Assume that there exist a closed petite set $M \in \mathcal{B}(\mathcal{X})$, a non-negative measurable function F bounded on M, a function $f \ge 1$, and constants $d, c > 0$ such that:*

$$\mathcal{A}F(x) \le -cf(x) + d1_M(x), \quad \forall x \in \mathcal{X}.$$

Then Φ is positive Harris recurrent.

5.2.2 Proof of Theorem 5.2

The state space of the Markov process $(\boldsymbol{n}(t))_{t\ge 0}$ is

$$\mathcal{X} = \bigcup_{n_{[1-K]} \in \mathbb{N}^K} \mathcal{N}(n_{[1-K]}).$$

Since $\mathcal{X}$ is countable, for all finite n_0, it is easy to see that $B_{n_0} = \{\boldsymbol{n} \in \mathcal{X} : n \le n_0\}$ is petite. In what follows, we will show that there exist F and a finite integer n_0 such that the drift $\mathcal{A}F(\boldsymbol{n})$ satisfies the condition of Theorem 5.3 with $M = B_{n_0}$.

The maximum social welfares for the user population of $\boldsymbol{n} \in \mathcal{N}(n_{[1-K]})$ are defined by: $V^\star(\boldsymbol{n}) = \max_{\boldsymbol{x} \in \mathcal{N}^R(n_{[1-K]})} V(\boldsymbol{x})$ and $W^\star(\boldsymbol{n}) = \max_{\boldsymbol{x} \in \mathcal{N}^R(n_{[1-K]})} W(\boldsymbol{x})$. We denote by $d_k(\boldsymbol{n})$ the departure rate of class-k users: $d_k(\boldsymbol{n}) = \sum_{s\in\mathcal{S}} \frac{n_{ks}}{n_k}\frac{\mu_{ks}}{n^s}$, and $\boldsymbol{d}(\boldsymbol{n}) = (d_1(\boldsymbol{n}), \dots, d_K(\boldsymbol{n}))$. We also define utility function J as: for all $\boldsymbol{x} \in \mathbb{R}_+^K$, $J(\boldsymbol{x}, \boldsymbol{n}) = \sum_{k\in\mathcal{K}} n_k \log(x_k/n_k)$. Note that $J(\boldsymbol{d}(\boldsymbol{n}), \boldsymbol{n}) = V(\boldsymbol{n})$ and $J^\star(\boldsymbol{n}) := \max_{\boldsymbol{x}\in\mathcal{R}} J(\boldsymbol{x}, \boldsymbol{n}) = V^\star(\boldsymbol{n})$ (cf. Proposition 2 of [13]). We use the following Lyapunov function: $F(\boldsymbol{n}) = Y(\boldsymbol{n}) + Z(\boldsymbol{n})$ where

$$Y(\boldsymbol{n}) = \sum_{k\in\mathcal{K}} \frac{n_k^2}{\rho_k} \quad \text{and} \quad Z(\boldsymbol{n}) = \frac{2S}{\beta} \cdot \big(\psi_{\max}(\boldsymbol{n}) - \psi(\boldsymbol{n})\big).$$

We show that the drift $\mathcal{A}F(\boldsymbol{n})$ is negative when the number of users is sufficiently large.

Step 1. We first study the drift $\mathcal{A}Y(\boldsymbol{n})$.

$$\begin{aligned}\mathcal{A}Y(\boldsymbol{n}) &= \sum_{k\in\mathcal{K}} 2\frac{n_k}{\rho_k}(\rho_k - d_k(\boldsymbol{n})) + \sum_{k\in\mathcal{K}} \frac{\rho_k + d_k(\boldsymbol{n})}{\rho_k} \\ &\le \sum_{k\in\mathcal{K}} 2\frac{n_k}{\rho_k}(\rho_k - d_k(\boldsymbol{n})) + 2KS\xi \\ &= 2\nabla J(\boldsymbol{\rho},\boldsymbol{n})^T \cdot (\boldsymbol{\rho} - \boldsymbol{d}(\boldsymbol{n})) + 2KS\xi \\ &\overset{(a)}{\le} 2\Big(J(\boldsymbol{\rho},\boldsymbol{n}) - J(\boldsymbol{d}(\boldsymbol{n}),\boldsymbol{n})\Big) + 2KS\xi \\ &= 2\Big(V^\star(\boldsymbol{n}) - V(\boldsymbol{n})\Big) - 2\Big(J^*(\boldsymbol{n}) - J(\boldsymbol{\rho},\boldsymbol{n})\Big) + 2KS\xi \quad (10)\end{aligned}$$

where (a) comes from concavity of J.

Step 2. We now investigate the drift $\mathcal{A}Z(\boldsymbol{n})$. This drift has two components, one due to the movements of users between servers and another one due to an arrival or a departure of a user.

$$\begin{aligned}\mathcal{A}Z(\boldsymbol{n}) \le &- 2\Big(W^\star(\boldsymbol{n}) - W(\boldsymbol{n}) - S\log\big(e\xi(n+S)\big)\Big) \\ &+ \frac{4\mu_{\max}S^2}{\beta}\log(n\xi). \qquad (11)\end{aligned}$$

The first part in the r.h.s. in the above inequality comes from the movements of users, and is derived from Theorem 4.1. The second part is due to arrivals and departures: the maximum drift due to one arrival or one departure is $\log(n\xi)$ and the arrival and departure rate cannot exceed $S\mu_{\max}$. Hence the drift due to arrivals and departures is upper bounded by $\frac{4\mu_{\max}S^2}{\beta}\log(n\xi)$.

Step 3. We are now ready to estimate $\mathcal{A}F(\boldsymbol{n})$. For notational simplicity, we define

$$R(\boldsymbol{n}) = 2KS\xi + 2S\log\big(e\xi(n+S)\big) + \frac{4\mu_{\max}S^2}{\beta}\log(n\xi).$$

Then, from (10) and (11), we get:

$$\begin{aligned}\mathcal{A}F(\boldsymbol{n}) =& \mathcal{A}Y(\boldsymbol{n}) + \mathcal{A}Z(\boldsymbol{n}) \\ \le& 2\Big(V^\star(\boldsymbol{n}) - V(\boldsymbol{n})\Big) - 2\Big(W^\star(\boldsymbol{n}) - W(\boldsymbol{n})\Big) \\ &- 2\Big(J^*(\boldsymbol{n}) - J(\boldsymbol{\rho},\boldsymbol{n})\Big) + R(\boldsymbol{n}) \\ \le& - 2\Big(J^*(\boldsymbol{n}) - J(\boldsymbol{\rho},\boldsymbol{n})\Big) + R(\boldsymbol{n}). \qquad (12)\end{aligned}$$

Observe that, since there exists $\boldsymbol{r} \in \mathcal{R}$ and $\delta > 0$ such that $(1+\delta)\boldsymbol{\rho} < \boldsymbol{r}$,

$$J^\star(\boldsymbol{n}) \ge J((1+\delta)\boldsymbol{\rho},\boldsymbol{n}) = J(\boldsymbol{\rho},\boldsymbol{n}) + n\log(1+\delta). \qquad (13)$$

From (13) into (12), we obtain:

$$\mathcal{A}F(\boldsymbol{n}) \le -2n\log(1+\delta) + R(\boldsymbol{n}).$$

Finally, since $R(\boldsymbol{n}) = \Theta(\log n)$, there exists a finite n_0 and $d > 0$ such that for all $\boldsymbol{n} \in \mathcal{X}$,

$$\mathcal{A}F(\boldsymbol{n}) \le -2n\log(1+\delta) + R(\boldsymbol{n}) \le -1 + d1_{B_{n_0}}(\boldsymbol{n}).$$

□

6. MULTI-SERVER ALLOCATIONS

In this section, we consider scenarios where each user can open several connections to the various servers – her service rate is the sum of the rates achieved on her connections. In wireless networks, this arises when devices have several radio interfaces. As before, servers share their time fairly among their connections. Each user is assumed to open $m > 1$ connections.

We introduce the following notation. In closed systems, the set of users is $\mathcal{U} = \{1,\dots,n\}$ and we denote by $\mathcal{U}_k$ the set of class-k users. The system state is represented by a vector $\boldsymbol{m} = (m_{us})_{u\in\mathcal{U},s\in\mathcal{S}} \in \mathbb{N}^{n\times S}$, where m_{us} is the number of connections of user u to server s. When the numbers of users of the different classes are $n_{[1-K]}$, $\mathcal{M}(n_{[1-K]})$ denotes the set of possible system states, i.e., $\boldsymbol{m} \in \mathcal{M}(n_{[1-K]})$ if (i) $m_{us} > 0$ only if $s \in \mathcal{S}_{k(u)}$ where $k(u)$ is the class of user u, (ii) $\sum_s m_{us} = m$, and (iii) $\sum_{u\in\mathcal{U}} 1_{k(u)=k} = n_k$. Finally, m^s denotes the number of connections to server s in state $\boldsymbol{m}$. In state $\boldsymbol{m}$, the service rate of user u is $\sum_s \frac{m_{us}\mu_{k(u)s}}{m^s}$.

To devise an efficient and distributed load resampling strategy, we use game theoretical techniques as earlier, but with a key difference. If we consider a game where players correspond to users, then this game is similar to a multi-path routing game as considered in [20, 21]. Unfortunately, this game does not always admit a pure NE [21]. A simple way to circumvent this difficulty consists in considering a game played by the connections (each connection aims at selfishly maximizing its service rate). This game is the same as that studied in Section 4, except that n is now replaced by nm. The analysis made in Sections 4 and 5 can be directly adapted. Following this approach, the RLS algorithm is simply replaced by the so-called mRLS algorithm described below.

The mRLS algorithm. Each user has a Poisson clock ticking at rate β. When the clock of a class-k user ticks, this user selects one of her active connections uniformly at random, say from server s, and a server s' uniformly at random in $\mathcal{S}_k$. She migrates the selected connection to the new server if this would increase the service rate of the connection, i.e., if $\mu_{ks}/m^s < \mu_{ks'}/(m^{s'}+1)$.

6.1 Ordinal Potential and Social Welfare

If $n_{[1-K]}$ is fixed, the system dynamics $\boldsymbol{m}(t)$ under the mRLS algorithm may be interpret as those observed in a load balancing game where nm strategic connections try to switch servers to improve their service rates. $\boldsymbol{m} \in \mathcal{M}(n_{[1-K]})$ is a pure NE of the game if and only if: $\forall u \in \mathcal{U}, \forall s \in \mathcal{S}_{k(u)} : m_{us} > 0$, $\forall s' \in \mathcal{S}_{k(u)}$, $\frac{\mu_{k(u)s}}{m^s} \ge \frac{\mu_{k(u)s'}}{m^{s'}+1}$. As shown in Section 4, this game is an ordinal potential game. A possible ordinal potential function is defined by: for all $\boldsymbol{m} \in \mathcal{M}(n_{[1-K]})$,

$$\psi(\boldsymbol{m}) = -\sum_{s\in\mathcal{S}} \log(m^s!) + \sum_{s\in\mathcal{S}}\sum_{k\in\mathcal{K}_s}\sum_{u\in\mathcal{U}_k} m_{us}\log(\mu_{ks}).$$

Under the mRLS algorithm, the ordinal potential $\psi(\boldsymbol{m}(t))$ increases over time, and $\boldsymbol{m}(t)$ converges to a pure NE.

Once again, the proportionally fair allocation of connections to servers plays an important role in the analysis of the system dynamics. We define the social welfare of state $\boldsymbol{m}$ as:

$$W(\boldsymbol{m}) = \sum_{u\in\mathcal{U}} \log\left(\sum_s \frac{\mu_{k(u)s}m_{us}}{m^s}\right).$$

Note that this welfare is defined from the users' perspective, not from the connections' point of view. The proportionally fair allocation maximizes $W(\boldsymbol{m})$ over all $\boldsymbol{m} \in \mathcal{M}(n_{[1-K]})$. To analyze the efficiency of multi-server allocations, we take as a reference the proportionally fair allocation in the ideal scenario where each user controls a large number of connections, in which case the system state is represented by a vector $\boldsymbol{x} \in \mathcal{M}^R(n_{[1-K]})$ where

$$\mathcal{M}^R(n_{[1-K]}) = \Big\{ \boldsymbol{x} \in \mathbb{R}_+^{n\times S} : \forall u, x_{us} = 0 \text{ if } s \notin \mathcal{S}_{k(u)}, \sum_{s\in\mathcal{S}_{k(u)}} x_{us} = m, \sum_u 1_{k(u)=k} = n_k \Big\}$$

Let $\boldsymbol{x}^\star$ be the maximizer of $W(\boldsymbol{x})$ over $\boldsymbol{x} \in \mathcal{M}^R(n_{[1-K]})$, and $W^\star = W(\boldsymbol{x}^\star)$. We further introduce two different notions of social welfare that help our analysis: for $\boldsymbol{m} \in \mathcal{M}(n_{[1-K]})$,

$$V(\boldsymbol{m}) = \sum_k n_k \log(\sum_{u\in\mathcal{U}_k}\sum_{s\in\mathcal{S}_k} \frac{m_{us}\mu_{ks}}{n_k m^s}),$$

$$U(\boldsymbol{m}) = \sum_{u\in\mathcal{U}}\sum_{s\in\mathcal{S}_{k(u)}} m_{us}\log(\frac{\mu_{us}}{m^s}).$$

Note that U is the welfare defined from the connections' perspective and it corresponds to the welfare W defined in Section 4 but with nm users. As in Lemma 4.1, we can show that $\frac{1}{m}U(\boldsymbol{m}) \le W(\boldsymbol{m}) \le V(\boldsymbol{m})$, and that if $\boldsymbol{x}' \in \arg\max_{x\in\mathcal{M}^R(n_{[1-K]})} V(\boldsymbol{x})$, then $W^\star = W(\boldsymbol{x}') = V(\boldsymbol{x}') = \frac{1}{m}U(\boldsymbol{x}')$.

6.2 Efficiency of NEs, Convergence Rate, and Stability

The results presented below are direct applications of those derived in Section 4.

6.2.1 Closed Systems

Under the mRLS algorithm, the system converges to a pure NE. Let $(t_i)_{i\ge 1}$ be the increasing sequence of epochs at which one user has the opportunity to update her allocation under mRLS. Then:

THEOREM 6.1. *For all $i \ge 1$, under mRLS, the expected drift in the potential after the update taking place at time t_{i+1} satisfies:*

$$\begin{aligned}
&\mathbb{E}\big[\psi(\boldsymbol{m}(t_{i+1})) - \psi(\boldsymbol{m}(t_i)) | \boldsymbol{m}(t_i)\big] \\
&\quad \ge \frac{1}{mnS}\Big(mW^\star - U(\boldsymbol{m}(t_i)) - S\log\big(e\xi(mn+S)\big)\Big) \\
&\quad \ge \frac{1}{nS}\Big(W^\star - W(\boldsymbol{m}(t_i)) - \frac{S}{m}\log\big(e\xi(mn+S)\big)\Big). \qquad (14)
\end{aligned}$$

From the above theorem, we may quantify the efficiency of pure NEs:

COROLLARY 6.1. *If $\boldsymbol{m}$ is a pure NE, then:*

$$W^\star - W(\boldsymbol{m}) \le \frac{S}{m}\log\big(e\xi(mn+S)\big).$$

Thus when users control more than one connection, the pure NEs become more efficient. Note that when m grows large, any pure NE $\boldsymbol{m}$ is efficient as $W^\star - W(\boldsymbol{m})$ tends to 0. However, as stated in the theorem below, it also takes more time to converge to these equilibriums.

THEOREM 6.2. *For any initial condition $\boldsymbol{n}(0)$, under the mRLS algorithm, we have for all t:*

$$\mathbb{E}\left[W^* - W(\boldsymbol{n}(t))\right] \le \frac{mnS}{\beta t}\log(S\xi) + \frac{S}{m}\log\big(e\xi mn(mn+S)\big).$$

6.2.2 Open Systems

When the user population evolves over time, we may use the results derived for closed systems to analyze the stability of the system under mRLS. The analysis is the same as that presented in Section 5. Again, mRLS turns out to be throughput optimal. With arrivals and departures, under the mRLS algorithm, the system state $\boldsymbol{m}(t)$ is a continuous-time Markov chain with countable state space $\cup_{n_{[1-K]}\in\mathbb{N}^K}\mathcal{M}(n_{[1-K]})$.

THEOREM 6.3. *(Throughput Optimality of mRLS) Assume that there exists $\boldsymbol{r} \in \mathcal{R}$ such that $\boldsymbol{\rho} < \boldsymbol{r}$, then under the mRLS algorithm, the Markov process $(\boldsymbol{m}(t))_{t\ge 0}$ is positive Harris recurrent.*

Note that the maximal stability region in the case where users control several connections remains $\mathcal{R}$. However, when the load of the system is relatively low, controlling m connections improves the average sojourn time (in very lightly loaded systems, this sojourn time is divided by a factor m provided that $m \le S_k, \forall k$).

7. CONTINUOUS ALLOCATIONS

We have seen that when each user controls m connections, a natural extension of RLS is shown to be throughput optimal. Opening m connections improves the system performance in light or moderate load scenarios. Observe however that when the user population is finite, there is still a gap between the socially optimal allocation and that realized under mRLS (see Theorem 6.1).

To close this gap, we investigate a scenario where users control an infinite number of connections ($m = \infty$). We may interpret such a scenario as follows: Users have a constant budget, say equal to 1, and bid on the various servers to obtain resources of these servers. We use the notation of the previous section. The system state is represented by a vector $\boldsymbol{w} = (w_{us})_{u\in\mathcal{U},s\in\mathcal{S}} \in \mathbb{R}_+^{n\times S}$, where w_{us} is the weight or bid of user u on server s. When the numbers of users of the different classes are $n_{[1-K]}$, $\mathcal{W}(n_{[1-K]})$ denotes the set of possible states, i.e., $\boldsymbol{w} \in \mathcal{W}(n_{[1-K]})$ if (i) $w_{us} > 0$ only if $s \in \mathcal{S}_{k(u)}$, (ii) $\sum_s w_{us} = 1$ for all u, s, and (iii) $\sum_{u\in\mathcal{U}} 1_{k(u)=k} = n_k$. Finally $w^s = \sum_{u\in\mathcal{U}} w_{us}$ is the total weight at server s. Servers fairly share their service capacity among users, and in state $\boldsymbol{w}$, the rate of users is $\sum_s \frac{w_{us}\mu_{k(u)s}}{w^s}$. This allocation can be seen as the limiting allocation considered in Section 6 when $m \to \infty$, and is referred to as the weighted proportional allocation [22].

7.1 Social Welfares

As in the previous section, we define the following social welfares:

$$\begin{aligned}
W(\boldsymbol{w}) &= \sum_{u\in\mathcal{U}} \log(\sum_{s\in\mathcal{S}_{k(u)}} \frac{\mu_{k(u)s}w_{us}}{w^s}) \\
V(\boldsymbol{w}) &= \sum_{k\in\mathcal{K}} n_k \log\left(\sum_{u\in\mathcal{U}_k}\sum_{s\in\mathcal{S}_k} \frac{\mu_{ks}w_{us}}{w^s n_k}\right) \\
U(\boldsymbol{w}) &= \sum_{u\in\mathcal{U}}\sum_{s\in\mathcal{S}_{k(u)}} w_{us}\log(\frac{\mu_{k(u)s}}{w^s}).
\end{aligned}$$

W captures users' utility, V corresponds to a per-class utility, and U is the social utility of a "non-atomic" system (each infinitesimal quantity composing the weight w_{us} is considered as a player). Denote by $\boldsymbol{w}^\star \in \mathcal{W}(n_{[1-K]})$ the weights maximizing W, and $W^\star = W(\boldsymbol{w}^\star)$. The following lemma states that W, V, and U achieve their maximum at the same point, so that to maximize W, we just need to either identify a maximizer for V or U.

LEMMA 7.1. *(i) For all $\boldsymbol{w}$, $V(\boldsymbol{w}) \geq W(\boldsymbol{w}) \geq U(\boldsymbol{w})$.*
(ii) Let $\boldsymbol{w}' \in \arg\max_{\boldsymbol{w}\in\mathcal{W}(n_{[1-K]})} U(\boldsymbol{w})$, then $\boldsymbol{x}'$ also maximizes V and W.

7.2 The Water-Filling Algorithm

Assume that the user population is fixed and described by $n_{[1-K]}$. Further assume that the system state is $\boldsymbol{w} = (w_u, \boldsymbol{w}_{-u})$, where w_u and $\boldsymbol{w}_{-u}$ represent the weights of user u and of all other users, respectively. As earlier, we propose a distributed load resampling scheme that corresponds to users' selfish moves. Again, if users update their weights so as to maximize their service rates, then the system dynamics become difficult to analyze. Instead when a user, say of class k, has the opportunity to update her weights, she does it so as to solve the convex program:

$$\max_{w_u} \sum_{s\in\mathcal{S}_{k(u)}} (w_{us} + w_{-u}^s)\log\left(\frac{\mu_{k(u)s}}{w_{us}+w_{-u}^s}\right)$$
$$\text{s.t.} \sum_{s\in\mathcal{S}_{k(u)}} w_{us} = 1,$$

where w_{-u}^s denotes the sum of the weights that all users except u put on server s. We can readily see that the above optimization problem is solved using a *water-filling* procedure:

$$\forall s, \quad w_{us} = \max\left\{0, \mu_{k(u)s}\lambda - w_{-u}^s\right\}, \tag{15}$$

where the water level λ is uniquely defined by $\sum_{s\in\mathcal{S}_k} w_{us} = 1$ (the Lagrange multiplier corresponding to the constraint $\sum_s w_{us} = 1$ is given by $\log\lambda + 1$).

The Water-Filling (WF) algorithm. Each user has a Poisson clock ticking at rate β. When the clock of a class-k user ticks, this user updates her weights according to (15).

7.3 Efficiency, Convergence Rate, and Stability

7.3.1 Closed Systems

Under the WF algorithm, the system state converges to an allocation maximizing the social welfare U. Indeed when user u updates her weights, the welfare U increases, for the components in U that depend on w_{us}, $s \in \mathcal{S}_{k(u)}$ reduce to:

$$\sum_{s\in\mathcal{S}_{k(u)}} w_{us}\log(\frac{\mu_{k(u)s}}{w^s}) + \sum_{u'\neq u}\sum_{s\in\mathcal{S}_{k(u)}} w_{u's}\log(\frac{1}{w^s})$$
$$= \sum_{s\in\mathcal{S}_{k(u)}} \left(w^s\log(\frac{1}{w^s}) + w_{us}\log(\mu_{k(u)s})\right)$$
$$= \sum_{s\in\mathcal{S}_{k(u)}} \left(w^s\log(\frac{\mu_{k(u)s}}{w^s}) - w_{-u}^s\log(\mu_{k(u)s})\right).$$

Hence in view of Lemma 7.1, WF converges to an allocation maximizing W, i.e., to the proportionally fair allocation. This result was expected (see Theorem 6.1 with $m \to \infty$).

LEMMA 7.2. *Under the WF algorithm, $\boldsymbol{w}(t)$ converges to $\boldsymbol{w}$ such that $W(\boldsymbol{w}) = W^\star$.*

We also establish that the rate at which the system converges is similar to that at which the system converges under RLS (users control a single connection). This is due to the fact that each user simultaneously updates her weights to all servers, which contrasts with mRLS where only one connection can be updated at a time. Let $(t_i)_{i\geq 1}$ be the increasing sequence of epochs at which one user has the opportunity to update her allocation under the WF algorithm. Then:

		Server 1	Server 2	Server 3	Server 4	Server 5
Class	1	5	3	1	0.1	0.1
Class	2	0.1	5	3	1	0.1
Class	3	0.1	0.1	5	3	1
Class	4	1	0.1	0.1	5	3
Class	5	3	1	0.1	0.1	5

Table 1: Service speeds

THEOREM 7.1. *For all $i \geq 1$, under WF, the expected drift in the utility U after the update taking place at time t_{i+1} satisfies:*

$$\mathbb{E}\big[U(\boldsymbol{w}(t_{i+1})) - U(\boldsymbol{w}(t_i))|\boldsymbol{w}(t_i)\big]$$
$$\geq \frac{1}{n}\Big(U^\star - U(\boldsymbol{w}(t_i)) - 2S\log\big(e\xi(n+S)\big)\Big). \tag{16}$$

THEOREM 7.2. *For any initial condition $\boldsymbol{w}(0)$, under the WF algorithm, we have for all t:*

$$\mathbb{E}\left[W^* - W(\boldsymbol{w}(t))\right] \leq \frac{n}{\beta t}\log(S\xi) + 2S\log\big(e\xi n(n+S)\big).$$

The proofs of the above theorems are similar to those of Theorems 4.1 and 4.2, and are presented in appendix for completeness.

7.3.2 Open System

The water-filling algorithm is throughput-optimal: under WF, $\boldsymbol{w}(t)$ is a continuous time Markov process with state space:

$$\mathcal{X} = \bigcup_{n_{[1-K]}\in\mathbb{N}^K} \mathcal{W}(n_{[1-K]}).$$

We show that if $\boldsymbol{\rho}$ is in $\mathcal{R}$ (note again that the maximal stability region remains equal to $\mathcal{R}$), $(\boldsymbol{w}(t))_{t\geq 0}$ is positive Harris recurrent. The state space $\mathcal{X}$ is not countable, which complicates the analysis. We introduce a metric on $\mathcal{X}$ so that the latter is a locally compact and separable metric space and use Theorem 5.3 to show stability.

THEOREM 7.3. *(Throughput Optimality of WF) Assume that there exists $\boldsymbol{r} \in \mathcal{R}$ such that $\boldsymbol{\rho} < \boldsymbol{r}$, then under the WF algorithm, the Markov process $(\boldsymbol{w}(t))_{t\geq 0}$ is positive Harris recurrent.*

The proof of Theorem 7.3 is similar to the proof of Theorem 5.2. Please refer to Appendix B.3.

8. NUMERICAL EXPERIMENTS

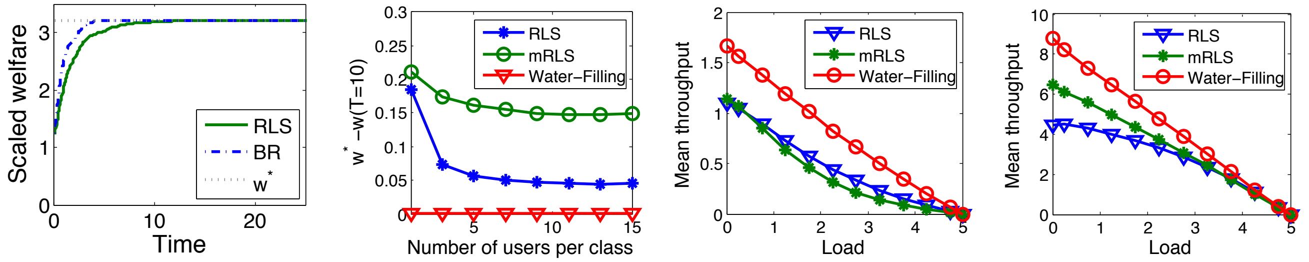
In this section, we present numerical experiments to illustrate our theoretical results using two examples of simple systems.

8.1 Case 1: A Toy Example

We first consider a toy example with 5 servers and 5 user classes. The service speeds μ_{ks} are given in Table 1. We set the number of connections $m = 2$ in the mRLS algorithm.

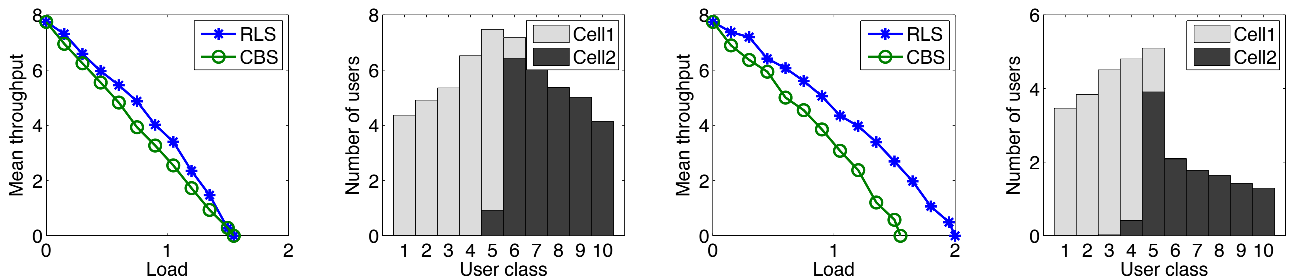
8.1.1 Closed System

We fix $\beta = 1$. In Fig. 1(a), we plot the evolution over time of the scaled social welfare w under BR and RLS algorithms for an initial system state where $n_{ks} = 5$ for $k \in \mathcal{K}$ and $s \in \mathcal{S}$. Both algorithms converge to an allocation that is approximately optimal, and RLS takes roughly twice as much time as BR to converge.



(a) Scaled social welfare vs. time (b) Social welfare gap at $t = 10$ (c) Mean throughput ($\beta = 1$) (d) Mean throughput ($\beta = 100$)

Figure 1: [Results for Case 1] Closed systems (a) and (b); Open systems (c) and (d).



(a) Mean throughput ($\beta = 10$) (b) Average number of users ($\rho = 1.5$) under RLS (c) Mean throughput ($\beta = 10$) (d) Average number of users ($\rho = 1.95$) under RLS

Figure 2: [Results for Case 2] Homogeneous loads (a) and (b); Heterogeneous loads (c) and (d).

In Fig. 1(b), we compare the RLS, mRLS, and WF algorithms. We present the gap in terms of scaled welfare under the three algorithms at time 10 as a function of the number of users per class (each class has the same number of users). This gap is equal to $w^\star - w(t = 10)$, where $w^\star = \frac{1}{n}(W^\star + n\log(n))$ with $W^\star = \max_{\boldsymbol{n}\in\mathcal{N}^R(n_{[1-K]})} W(\boldsymbol{n})$ and $w(t = 10)$ is $\frac{1}{n}(W(t = 10) + n\log(n))$. Note that the definition of W depends on the systems considered (one connection, multiple connections, or continuous connections), but $W^\star$ is common to all systems, e.g.
$W^\star = \max_{\boldsymbol{m}\in\mathcal{M}^R(n_{[1-K]})} W(\boldsymbol{m})$. In the initial state, users are spread randomly over all servers. Under the WF algorithm, users find the best allocation rapidly; it converges to the optimal allocation before $t = 10$ with high probability, whereas this is not the case for RLS and mRLS. On the other hand, since only one connection may move at each update under mRLS, its convergence rate is proportional to $1/m$ and thus the convergence of mRLS is slower than that of RLS (see Theorem 6.2). Also observe that the gap under RLS and mRLS decreases as the number of users per class increases. This illustrates Theorems 4.2 and 6.2, i.e., the pure NEs may be inefficient when the number of users is small.

8.1.2 *Open System*

For open systems, we assume that the service requirements are i.i.d. exponentially distributed with unit mean. Users of the various classes arrive according to Poisson processes. The performance of the algorithms are illustrated using the *mean throughput*, equal to the inverse of the average sojourn time of users. We consider a symmetric scenario, where $\rho_k = \rho$ for all k.

Fig. 1(c) and 1(d) present the mean throughput as a function of the load ρ under RLS, mRLS, and WF when $\beta = 1$ and $\beta = 100$, respectively. First observe that as expected, all algorithms are throughput optimal. They are stable when the load is smaller than 5 – as if all users were always served at maximum speed 5.

Also note that β has a strong impact on performance. In lightly loaded systems, the mean throughput increases with the rate at which users may switch servers. When β is very large, each user moves frequently and rapidly finds the best server. For example, when ρ_k is close to 0, if $\beta = 100$, under RLS, users are served at rate almost equal to 5 in average. In contrast, when β is small, users may well not move before they leave the system, which explains their low mean throughput, see Fig. 1(c). The impact of β is here exacerbated by the large difference between the service speeds at the worst and best servers (from 0.1 to 5).

In general opening multiple connections increases the performance, especially in lightly loaded systems. In this case, a single user can simultaneously exploit the resources of multiple servers. The performance gain typically decreases with the load, and tends to vanish in heavy traffic. It should be observed that mRLS may well yield lower performance than RLS. Again, this is due to the fact that mRLS only moves at most one connection at each update – the benefit of having multiple connections is mitigated by the low convergence rate. Note that under WF, the convergence is quick, and users always exploit multiple resources of multiple servers. WF always provides the best quality of service.

8.2 Case 2: A Cellular System

In wireless cellular networks, the channel capacities between the users and the base stations (the servers) depend on their respective locations. Users close the base station experience a better channel quality, and are served at a high rate. Thus, in general, it makes sense to attach users to the closest base station (BS), an algorithm referred to as “Closest BS Selection (CBS)”. In this section, we compare RLS with CBS in a simple scenario. Consider a network with two base stations and 10 user classes. We denote by $\mathcal{S} = \{1, 2\}$ and $\mathcal{K} = \{1, \ldots, 10\}$ the sets of base stations and user classes, respectively. We define the service speed of class-k users as follows: $\mu_{k1} = 11 - k$ at cell 1 and $\mu_{k2} = k$ at cell 2. Then, under CBS, class-k users select BS 1 if $1 \le k \le 5$ and the other users select BS 2.

We investigate open systems only. When users of the various classes have the same arrival rate (homogeneous loads), CBS and RLS exhibit very similar performance. Fig. 2(a) shows the mean throughput of the system, where $\rho_k = \rho$ for all k. Both CBS and RLS stabilize the system as long as $\rho < 1.55$ (this is the maximal load compatible with stability $1/(1/10+1/9+1/8+1/7+1/6) \simeq 1.55$). RLS performs slightly better than CBS. This little gain comes from the fact that when one of the base stations becomes idle, RLS can utilize its resource by balancing the load. Since the arrival rates are symmetric, almost all users of class $k \in \{1, \ldots, 5\}$ stay in cell 1 and users of class $k' \in \{6, \ldots, 10\}$ in cell 2 even under RLS, which is captured in Fig. 2(b).

When user arrival pattern is asymmetric, CBS performance becomes poor. Consider the heterogeneous scenario where $\rho_k = \rho$ for $1 \le k \le 5$ and $\rho_k = \rho/2$ for $6 \le k \le 10$. Then, under CBS, the overall load in cell 1 is 2 times higher than that in cell 2. Fig. 2(c) presents the mean throughput in this scenario. RLS can stabilize the system as long as $\rho < 2$ while the mean throughput of CBS is 0 for $\rho > 1.55$, i.e., it is unstable. Under RLS, the loads on two cells get balanced by the migration of class-5 users, which is shown in Fig. 2(d) where about 80% of class-5 users are in cell 2 although $\mu_{51} > \mu_{52}$.

9. CONCLUSION

In this paper, we have analyzed the performance of the RLS algorithm, a very simple distributed load resampling strategy for heterogenous parallel server systems. In scenarios where the user population is fixed, we were able to quantify at any time the efficiency of the allocation of users to servers under this algorithm. Using this result, we proved that when users arrive and leave the system upon service completion, RLS is throughput optimal, i.e., it stabilizes the system whenever this is at all possible. In other words, RLS is able to fully exploit the system heterogeneity when the user population is large, which is quite surprising.

There are many interesting directions for future work. It would be for example useful to derive estimates of the mean user sojourn times under the RLS algorithm and its extensions. This could be done by studying the system in various limiting regimes. We can use mean field asymptotics to analyze systems with a large number of servers. We may also look at systems in heavy traffic.

10. REFERENCES

[1] A. Ganesh, S. Lilienthal, D. Manjunath, A. Proutiere, and F. Simatos, “Load balancing via random local search in closed and open systems,” in *Proc. ACM SIGMETRICS*, 2010, pp. 287–298.

[2] N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani, *Algorithmic Game Theory*. New York, NY, USA: Cambridge University Press, 2007.

[3] E. Koutsoupias and C. Papadimitriou, “Worst-case equilibria,” in *Proc. STACS*, 1999, pp. 404–413.

[4] A. Orda, R. Rom, and N. Shimkin, “Competitive routing in multiuser communication networks,” *IEEE/ACM Trans. Netw.*, vol. 1, no. 5, pp. 510–521, Oct. 1993.

[5] P. Berenbrink, T. Friedetzky, I. Hajirasouliha, and Z. Hu, “Convergence to equilibria in distributed, selfish reallocation processes with weighted tasks,” *Algorithmica*, vol. 62, no. 3-4, pp. 767–786, 2012.

[6] I. Milchtaich, “Congestion games with player-specific payoff functions,” *Games and Economic Behavior*, vol. 13, no. 1, pp. 111 – 124, 1996.

[7] V. Anantharam, “On the nash dynamics of congestion games with player-specific utility,” in *Proc. IEEE CDC*, vol. 5, 2004, pp. 4673–4678 Vol.5.

[8] J. Ackermann, “Pure nash equilibria in player-specific and weighted congestion games,” *Theoretical Computer Science*, vol. 410, no. 17, pp. 1552 – 1563, 2009.

[9] M. Gairing and M. Klimm, “Congestion games with player-specific costs revisited,” in *Proc. SAGT*, 2013, pp. 98–109.

[10] D. Shah and J. Shin, “Dynamics in congestion games,” in *Proc. ACM SIGMETRICS*, 2010, pp. 107–118.

[11] A. L. Stolyar, “Optimal routing in output-queued flexible server systems,” *Probab. Eng. Inf. Sci.*, vol. 19, no. 2, pp. 141–189, Apr. 2005.

[12] R. Etkin, A. Parekh, and D. Tse, “Spectrum sharing for unlicensed bands,” *IEEE JSAC*, vol. 25, no. 3, pp. 517–528, 2007.

[13] S. Borst, A. Proutiere, and N. Hegde, “Capacity of wireless data networks with intra- and inter-cell mobility,” in *Proc. IEEE INFOCOM*, 2006, pp. 1–12.

[14] G. Kasbekar and A. Proutiere, “Opportunistic medium access in multi-channel wireless systems: A learning approach,” in *Proc. Allerton*, 2010.

[15] D. Monderer and L. S. Shapley, “Potential games,” *Games and Economic Behavior*, vol. 14, no. 1, pp. 124 – 143, 1996.

[16] M. Voorneveld and H. Norde, “A characterization of ordinal potential games,” *Games and Economic Behavior*, vol. 19, no. 2, pp. 235–242, May 1997.

[17] F. Kelly, A. Maulloo, and D. Tan, “Rate control for communication networks: shadow prices, proportional fairness and stability,” *Journal of the Operational Research Society*, no. 3, pp. 237–252, 1998.

[18] F. Simatos and D. Tibi, “Spatial homogenization in a stochastic network with mobility,” *Ann. Appl. Probab.*, vol. 20, no. 1, pp. 312–355, 2010.

[19] S. P. Meyn and R. L. Tweedie, “Stability of Markovian processes III: Foster-Lyapunov criteria for continuous-time processes,” *Advances in Applied Probability*, vol. 25, no. 3, pp. 518–548, Sep. 1993.

[20] P. Key, L. Massoulié, and D. Towsley, “Path selection and multipath congestion control,” *Commun. ACM*, vol. 54, no. 1, pp. 109–116, Jan. 2011.

[21] P. Key and A. Proutiere, “Routing games with elastic traffic,” *SIGMETRICS Perform. Eval. Rev.*, vol. 37, no. 2, pp. 63–64, Oct. 2009.

[22] T. Nguyen and M. Vojnovic, “Weighted proportional allocation,” in *Proc. ACM SIGMETRICS*, 2011, pp. 173–184.

APPENDIX

A. PROOFS OF LEMMAS

A.1 Proof of Lemma 4.1

(i) By concavity of the log function:

$$\log\left(\sum_{s\in\mathcal{S}}\frac{n_{ks}\mu_{ks}}{n_k n^s}\right)\geq\sum_{s\in\mathcal{S}}\frac{n_{ks}}{n_k}\log(\frac{\mu_{ks}}{n^s}),$$

which implies $V(\boldsymbol{n})\geq W(\boldsymbol{n})$.

(ii) It was shown in [13] that V is maximized at $\boldsymbol{x}$ when the following conditions hold: a. when $x_{ks}>0$, $\frac{\mu_{ks}}{n^s}=\frac{\mu_{ks'}}{n^{s'}}$ if $x_{ks'}>0$; b. when $x_{ks}=0$. $\frac{\mu_{ks}}{n^s}\leq\frac{\mu_{ks'}}{n^{s'}}$ if $x_{ks'}>0$. For $\boldsymbol{x}$ satisfying these two conditions, it is clear that $W(\boldsymbol{x})=V(\boldsymbol{x})$, and hence in view of (i), $\boldsymbol{x}$ also maximizes W. □

A.2 Proof of Lemma 7.1

(i) Since log is a concave function,

$$\log\left(\sum_{s\in\mathcal{S}_{k(u)}}w_{us}\frac{\mu_{k(u)s}}{w^s}\right)\geq\sum_{s\in\mathcal{S}_{k(u)}}w_{us}\log(\frac{\mu_{k(u)s}}{w^s}).$$

Therefore, $W(\boldsymbol{w})\geq U(\boldsymbol{w})$.

(ii) In (i), the equality holds when every class k satisfies that

$$\frac{\mu_{ks}}{w^s}=\frac{\mu_{ks'}}{w^{s'}}$$

for all s and s' having positive weight $w_{ks},\ w_{ks'}>0$. This is the KKT condition of V and W. □

B. PROOFS OF THEOREMS

B.1 Proof of Theorem 7.1

The proof is similar to that of Theorem 4.1.

$$\begin{aligned}
&n\mathbb{E}\left[U(\boldsymbol{w}(t_{i+1}))-U(\boldsymbol{w}(t_i))|\boldsymbol{w}(t_i)=\boldsymbol{w}\right]\\
&\geq\sum_{u\in\mathcal{U}}\sum_{s\in\mathcal{S}_{k(u)}}w_{us}\max\{0,\max_{c\in\mathcal{S}_{k(u)}}\log\frac{\mu_{k(u)c}}{w^c+1}-\log\frac{\mu_{k(u)s}}{w^s-1}\}\\
&\geq\sum_{u\in\mathcal{U}}\sum_{s\in\mathcal{S}_{k(u)}}w_{us}\left(\max_{c\in\mathcal{S}_{k(u)}}\log\frac{\mu_{k(u)c}}{w^c+1}-\log\frac{\mu_{k(u)s}}{w^s-\delta^*_{k(u)s}}\right)\\
&=\sum_{u\in\mathcal{U}}\sum_{s\in\mathcal{S}_{k(u)}}w_{us}\left(\max_{c\in\mathcal{S}_{k(u)}}\log\frac{\mu_{k(u)c}}{w^c+1}-\log\frac{\mu_{k(u)s}}{w^s}\right)\\
&\qquad-\sum_{u\in\mathcal{U}}\sum_{s\in\mathcal{S}_{k(u)}}w_{us}\log\frac{w^s}{w^s-\delta^*_{k(u)s}}\\
&\overset{(a)}{\geq}U^\star-U(\boldsymbol{w})-S\log(e\xi(n+S))-S-S\log(\xi n),
\end{aligned}$$

where

$$\delta^*_{ks}=\arg\min_{\delta\in[0,1]}\left|\max_{c\in\mathcal{S}_k}\log\frac{\mu_{kc}}{w^c+1}-\log\frac{\mu_{ks}}{w^s-\delta^*_{ks}}\right|.$$

It remains to prove inequality (a). As in the proof of Theorem 4.1, we can easily show that:

$$\begin{aligned}
&\sum_{u\in\mathcal{U}}\sum_{s\in\mathcal{S}_{k(u)}}w_{us}\left(\max_{c\in\mathcal{S}_{k(u)}}\log\frac{\mu_{k(u)c}}{w^c+1}-\log\frac{\mu_{k(u)s}}{w^s}\right)\\
&\qquad\qquad\geq U^\star-U(\boldsymbol{w})-S\log(e\xi(n+S)).
\end{aligned}$$

Moreover, since $\frac{1}{\xi}\leq w^s-\delta^*_{ks}\leq w^s$,

$$\max_{k\in\mathcal{K},s\in\mathcal{S}_k}\log\frac{w^s}{w^s-\delta^*_{ks}}\leq\log(\xi n).\tag{17}$$

Thus,

$$\begin{aligned}
\sum_{k\in\mathcal{K}}\sum_{s\in\mathcal{S}_k}w_{ks}\log\frac{w^s}{w^s-\delta^*_{ks}}&\leq S\log(\xi n)+\sum_{s\in\mathcal{S}}w'^s\log\frac{w^s}{w'^s}\\
&\leq S\log(\xi n)+S\log(e),
\end{aligned}$$

where $w'^s=\max\{0,w^s-1\}$. This completes the proof of inequality (a). □

B.2 Proof of Theorem 7.2

To prove the theorem, we first provide a lower and an upper bound of $U(\boldsymbol{w})$. For all $\boldsymbol{w}\in\mathcal{W}(n_{[1-K]})$, we have:

$$\begin{aligned}
U(\boldsymbol{w})=&\sum_{u\in\mathcal{U}}\sum_{s\in\mathcal{S}_{k(u)}}w_{us}\log\mu_{k(u)s}-\sum_{s\in\mathcal{S}}w^s\log(w^s)\\
\leq& n\mu_{\max}-\sum_{s\in\mathcal{S}}\frac{w^sS}{S}\log(w^sS)+n\log S\\
\leq& n\mu_{\max}-n\log(n)+n\log S
\end{aligned}\tag{18}$$

and

$$U(\boldsymbol{w})\geq n\mu_{\min}-\sum_{s\in\mathcal{S}}w^s\log(w^s)\geq n\mu_{\min}-n\log(n).\tag{19}$$

From (18) and (19),

$$U_{\max}-U_{\min}\leq n\log(\xi)+n\log S,\tag{20}$$

where we let $U_{\max}=\max_{\boldsymbol{w}\in\mathcal{W}(n_{[1-K]})}U(\boldsymbol{w})$ and $U_{\min}=\min_{\boldsymbol{w}\in\mathcal{W}(n_{[1-K]})}U(\boldsymbol{w})$.

Next, as in Lemma 4.3, we provide an upper bound of the minimum average gap between the maximum welfare $U^\star$ and the welfare after $i\leq I$ updates under the WF algorithm. From Theorem 7.1, we have:

$$\begin{aligned}
&\mathbb{E}\left[U(\boldsymbol{w}(t_{I+1}))-U(\boldsymbol{w}(0))\right]\\
&=\sum_{i=0}^{I}\mathbb{E}\left[U(\boldsymbol{w}(t_{i+1}))-U(\boldsymbol{w}(t_i))\right]\\
&\geq\sum_{i=0}^{I}\mathbb{E}\left[\frac{1}{n}\Big(U^\star-U(\boldsymbol{w}(t_i))-2S\log\big(e\xi(n+S)\big)\Big)\right].
\end{aligned}\tag{21}$$

From (20) and (21),

$$\begin{aligned}
&\min_{i\leq I}\mathbb{E}[U^\star-U(\boldsymbol{w}(t_i))]\\
&\leq\frac{n}{I+1}\mathbb{E}[U(\boldsymbol{w}(t_{I+1}))-U(\boldsymbol{w}(0))]+2S\log(e\xi(n+S))\\
&\leq\frac{n}{I+1}n\log(S\xi)+2S\log(e\xi(n+S)).
\end{aligned}\tag{22}$$

We are now ready to prove Theorem 7.2. Let τ be the number of times up to time t users get an opportunity to update their weights under the WF algorithm. From (22), given $\tau=I$, there exists $i^\star\leq I$ such that:

$$\begin{aligned}
&\mathbb{E}\left[U^\star-U(\boldsymbol{w}(t_{i^\star}))|\tau=I\right]\\
&\leq\frac{n^2}{I+1}\cdot\log(S\xi)+2S\log(e\xi(n+S)),
\end{aligned}$$

and, since, for all i, $U(\boldsymbol{w}(t_{i+1})) - U(\boldsymbol{w}(t_{i+1})) \geq 0$,

$$\begin{aligned}\mathbb{E}\left[U^\star - U(\boldsymbol{w}(t)) | \tau = I\right] &\leq \mathbb{E}\left[U^\star - U(\boldsymbol{w}(t_{i^*}))\right] \\ &\leq \frac{n^2}{I+1}\log(S\xi) + 2S\log(e\xi(n+S)).\end{aligned}$$

Now, since τ has a Poisson distribution with mean $\lambda = \beta n t$, we have $\mathbb{E}[\frac{1}{I+1}] = \sum_{I=0}^{\infty} \frac{\lambda^I e^{-\lambda}}{I!(I+1)} \leq \frac{1}{\lambda}$.
Finally, we obtain:

$$\begin{aligned}\mathbb{E}\left[W^\star - W(\boldsymbol{w}(t))\right] &\leq \mathbb{E}\left[U^\star - U(\boldsymbol{w}(t))\right] \\ &\leq \frac{n}{\beta t}\log(S\xi) + 2S\log(e\xi(n+S)).\end{aligned}$$

□

B.3 Proof of Theorem 7.3

Let $\boldsymbol{x} \in \mathcal{X}$. We denote by $n(\boldsymbol{x})$ the corresponding number of users, by $n_k(\boldsymbol{x})$ the number of class-k users, and by $n_{[1-K]}(\boldsymbol{x}) = (n_1(\boldsymbol{x}), \cdots, n_K(\boldsymbol{x}))$. Users are indexed by $u = 1, \dots, n(\boldsymbol{x})$, and $k_{\boldsymbol{x}}(u)$ the class of user u. We introduce the following metric on $\mathcal{X}$:

$$\begin{aligned}d(\boldsymbol{x}, \boldsymbol{y}) = &\sum_{u=1}^{\min\{n(\boldsymbol{x}), n(\boldsymbol{y})\}} \sum_{s \in \mathcal{S}} |x_{us} - y_{us}| \cdot 1_{k_{\boldsymbol{x}}(u) = k_{\boldsymbol{y}}(u)} \\ &+ \sum_{u=1}^{\min\{n(\boldsymbol{x}), n(\boldsymbol{y})\}} \sum_{s \in \mathcal{S}} 2 \cdot 1_{k_{\boldsymbol{x}}(u) \neq k_{\boldsymbol{y}}(u)} + 2|n(\boldsymbol{x}) - n(\boldsymbol{y})|.\end{aligned}$$

Note that, in this metric, the order in which users appear in $\boldsymbol{x}$ matters may not seem natural. We could have defined a metric that does not depend on this order, but this is not needed here. Let $\mathbf{0}$ denote the empty state and define $\|\boldsymbol{x}\| = d(\boldsymbol{x}, \mathbf{0})$. Then $\mathcal{X}$ is a locally compact and separable metric space. The Markov process $(\boldsymbol{w}(t))_{t \geq 0}$ takes its values in $\mathcal{X}$, and has transition rates such that when there is an arrival or a departure, users are relabeled randomly. When a user updates her weights, we do not change the labels of users. The precise order in which users are labeled in the construction of the Markov process $(\boldsymbol{w}(t))_{t \geq 0}$ does not really matter: in fact to study its positive Harris recurrence, we use a Lyapunov function whose drift does not depend on the order in which users are considered in the state.

Let us first identify a closed petite set:

LEMMA B.1. *Let $n_0 > 0$. The set $C_{n_0} : \{\boldsymbol{x} \in \mathcal{X} : \|\boldsymbol{x}\| \leq n_0\}$ is a closed petite set.*

Proof. Let $\boldsymbol{x} \in C_{n_0}$. Then, there is a strictly positive probability that the system starting in $\boldsymbol{x}$ is in state $\mathbf{0}$ at time $t = 1$ (we can construct paths with only departures – no moves, and no arrivals). Hence choosing for example $\nu(\cdot) = \delta_1(\cdot)$, we can prove that there exists $\varepsilon > 0$ such that $K_\nu(\boldsymbol{x}, \mathbf{0}) = P^1(\boldsymbol{x}, \mathbf{0}) > \varepsilon$ for all $\boldsymbol{x} \in C_{n_0}$. Therefore, C_{n_0} is ϕ_ν-petite where $\phi_\nu(A) = \varepsilon$ if $\mathbf{0} \in A$ and $\phi_\nu(A) = 0$ otherwise. □

LEMMA B.2. *Let $\mathcal{A}$ denote the generator of the Markov process $(\boldsymbol{w}(t))_{t \geq 0}$. Let $F(\boldsymbol{w}) = \sum_{k=1}^{K} \frac{n_k(\boldsymbol{w})^2}{\rho_k} + \frac{2}{\beta} \cdot \big(U^\star(\boldsymbol{w}) - U(\boldsymbol{w})\big)$. Then, there exists n_0 such that for some $a, b > 0$*

$$\mathcal{A}F(\boldsymbol{w}) \leq -a + b 1_{C_{n_0}}.$$

Proof. The proof is similar to that of Theorem 5.2. $\boldsymbol{d}$, Y and Z are defined as

$$d_k = d_k(\boldsymbol{w}) = \sum_{u \in \mathcal{U}_k} \sum_{s \in \mathcal{S}_k} \frac{w_{us}}{n_k(\boldsymbol{w})} \frac{\mu_{ks}}{w^s},$$

$$Y(\boldsymbol{w}) = \sum_{k \in \mathcal{K}} \frac{n_k(\boldsymbol{w})^2}{\rho_k} \quad \text{and} \quad Z(\boldsymbol{w}) = \frac{2}{\beta} \cdot \big(U^\star(\boldsymbol{w}) - U(\boldsymbol{w})\big),$$

where $U^\star(\boldsymbol{w}) = \max_{\boldsymbol{x} \in \mathcal{W}^R(n_{[1-K]}(\boldsymbol{w}))} U(\boldsymbol{x})$.

The analysis of the drift $\mathcal{A}Y(\boldsymbol{w})$ is the same as in the proof of Theorem 5.2:

$$\begin{aligned}\mathcal{A}Y(\boldsymbol{w}) - 2KS\xi &\leq 2\Big(J(\boldsymbol{\rho}, \boldsymbol{w}) - J(\boldsymbol{d}, \boldsymbol{w})\Big) \\ &= 2\Big(U^\star(\boldsymbol{w}) - V(\boldsymbol{w})\Big) - 2\Big(U^\star(\boldsymbol{w}) - J(\boldsymbol{\rho}, \boldsymbol{w})\Big), \quad (23)\end{aligned}$$

where J is defined as in the proof of Theorem 5.2: for all $\boldsymbol{x} \in \mathbb{R}_+^K$,

$$J(\boldsymbol{x}, \boldsymbol{w}) = \sum_{k \in \mathcal{K}} n_k(\boldsymbol{w}) \log(x_k / n_k(\boldsymbol{w})).$$

By Theorem 7.1, we deduce that the drift $\mathcal{A}Z(\boldsymbol{w})$ satisfies:

$$\begin{aligned}\mathcal{A}Z(\boldsymbol{w}) \leq &- 2\Big(U^\star(\boldsymbol{w}) - U(\boldsymbol{w})\Big) \\ &+ 4S\log\big(e\xi(n+S)\big) + \frac{4\mu_{\max}S}{\beta}\log(n\xi), \quad (24)\end{aligned}$$

where $n = n(\boldsymbol{w})$ (to simplify the notation). Therefore, if we define

$$R(\boldsymbol{w}) := 2KS\xi + 4S\log\big(e\xi(n+S)\big) + \frac{4\mu_{\max}S}{\beta}\log(n\xi),$$

we get:

$$\begin{aligned}\mathcal{A}F(\boldsymbol{w}) &\leq - 2\Big(U^\star(\boldsymbol{w}) - J(\boldsymbol{\rho}; \boldsymbol{n})\Big) + R(\boldsymbol{w}) \\ &\leq - 2n\log(1+\delta) + R(\boldsymbol{w}). \quad (25)\end{aligned}$$

We conclude exactly as in the proof of Theorem 5.2. □