Some applications of asymptotic analysis in communication networks

Balakrishna Prabhu

THÈSE D'HABILITATION À DIRIGER DES RECHERCHES

SOME APPLICATIONS OF ASYMPTOTIC ANALYSIS IN COMMUNICATION NETWORKS

Par

BALAKRISHNA PRABHU

SOUTENUE LE 10 NOVEMBRE 2023

MEMBRES DU JURY

PRÉSIDENTE :	JOHANNE COHEN	DIRECTRICE DE RECHERCHES, CNRS, UNIV. PARIS SACLAY
RAPPORTEURS :	SEM BORST	PROFESSOR, EINDHOVEN UNIVERSITY OF TECHNOLOGY
	SEAN MEYN	PROFESSOR, UNIVERSITY OF FLORIDA
	BRUNO GAUJAL	DIRECTEUR DE RECHERCHES, INRIA GRENOBLE
MEMBRES :	ESA HYYTIÄ	PROFESSOR, UNIVERSITY OF ICELAND
	BENNY VAN HOUDT	PROFESSOR, UNIVERSITY OF ANTWERP
	URTZI AYESTA	DIRECTEUR DE RECHERCHES, CNRS, IRIT-ENSEEIHT

TABLE OF CONTENTS

ACKNOWLEDGMENTS

First and foremost, I would like to express my gratitude to CNRS and, in particular, LAAS for offering me the opportunity to carry out research in extremely favorable conditions.

It is an honor for me to have Sem, Sean, and Bruno as reviewers of this manuscript, and Johanne, Esa, Benny, and Urtzi as examiners. My sincere thanks to all of them.

Research work is rarely a solitary affair. I have had the privilege to interact and collaborate with and learn from several colleagues over the years and especially after my PhD. Dieter, Eitan, Hannu, Ilkka, Jacques, JK, Koen, Kostia, Maaike, Manjuanth, Matthieu, Olivier, Onno, Rachid, Rhonda, Sindo, and Urtzi have all contributed to shaping and enriching my research in their own way and deserve all the credit.

Thanks are also due to my post-docs (Trung and Morgan) and PhD students (Tatiana, Josu, Josselin, Hang, Nga, and Iovi) for their contributions and the learning experience.

Finally, I would like to thank my family for their unflinching and invaluable support.

OVERVIEW

My research activities revolve around resource allocation in stochastic networks and in particular in those that can be modelled using Markovian dynamics. In a typical open stochastic-network model, objects enter the network in some node, move along a possibly random path and then leave the network at their destination node. These models find applications in a wide-range of domains such as communication networks, road-traffic networks, manufacturing facilities, and chemical-reaction networks. As an illustration, in communication networks, the objects can be packets that arrive at the source node, are sent over a series of one or more communication channels until they reach the destination where they exit the network. In road-traffic networks, the objects can be vehicles, while in manufacturing facilities, products can take on this role. Several interesting problems arise in this context including *dimensioning* which is about provisioning capacity of the nodes and the edges subject to budget constraints; *scheduling* which focuses on allocating capacity at a node or at an edge when multiple objects are competing for the same resource; and *routing* where one determines the paths that objects take within the network to reach the destination. All three problems (either individually or in some combination) can be viewed as optimization or control problems with an objective that depends usually in some non-obvious way upon the control variables.

Ideally, these three problems would be treated together. However, in many practical applications, even a separate treatment of each problem is not trivial. One reason for this is the size of the problem. Imagine the routing problem of allocating tasks to processors with the objective of minimizing the average weighted completion time of tasks. Here, the weight can be thought of as a penalty per unit time that has to be paid to the task to make it wait. The number of possible allocations grows exponentially in the number of objects which, in networks with thousands of objects and nodes, makes it computationally prohibitive to solve the optimization problem. Another reason for the difficulty is due to the lack of knowledge the parameters of the tasks. In practice, tasks arrive over time and have sizes that are unknown in advance and sometimes even at the allocation instants.

Several approaches have been employed to overcome these obstacles. One of them is to restrict attention to algorithms, which are recipes to obtain a solution, that are lower in complexity but suboptimal. If a guarantee (*e.g.*. worst-case ratio) on the quality of the solution can be exhibited, then the algorithms are known as approximation algorithms otherwise they are commonly referred to as heuristics. Approximation algorithms, which is a vibrant field in theoretical computer science with a rich history [1, 2], can be conceived for both the offline scenario in which there is complete knowledge of all the parameters as well as for the online scenario with only partial (current and past) knowledge. They are robust in the sense that their guarantees hold for the worst-case parameters. On the other hand, they have certain shortcomings. First, these guarantees are for finite-sized instances (number of tasks, *e.g.*) which makes it difficult to study notions such as stability which ensures that the number of objects in the network does not grow over time. As an example, consider a manufacturing facility with several machines. An approximation algorithm for scheduling the orders will only guarantee the mean delivery times with respect to the optimal schedule. If the facility is not well-dimensioned (not enough machines), then orders may build up over time leading to a growing delivery time even for the optimal schedule. The relative guarantee is still respected for all finite time instants but the delivery times will be so large that they will no longer be useful for the clients. If it were known how the capacity of the machines influences the delivery time, the facility could be appropriately dimensioned and estimated delivery times could be provided to the clients. Approximation algorithms do not usually concern themselves with quantifying the relationship between the objective and the input parameters. A second drawback is that worst-case analysis may be too pessimistic especially when the system designer may possess some statistics on the parameters that can be used to obtain finer and perhaps improved bounds. An algorithm with a very bad worst-case guarantee may nevertheless work well on average for this particular scenario as the worst-case instance may be highly unlikely. Such an algorithm may be preferred to one with a better worst-case guarantee but which has a worse average-case performance.

This bring us to the second approach based on stochastic models that exploits the knowledge of the statistics. The usual recipe is to model the number of objects

at each node as a dynamic process and investigate how the probability distribution of configurations (or states) evolves over time. A subclass of these models are governed by Markovian dynamics, that is, one in which the probability distribution in the future can be determined knowing only the current state. The past can therefore be forgotten allowing for a compact state-description. Markovian models can be used for analysis (*i.e.*, to compute formulas for the objective function) [3] as well as for optimal control using Markov decision process (MDPs) [4] over both finite and infinite horizons. Going back to the example of the manufacturing facility, with this approach one can figure out the values of the capacities for which the number of orders remain finite. Further, depending upon the model, it may be possible to determine the mean delivery time of the orders and how it depends upon the capacities. The manufacturer can the decide how much capacity to install in order not to keep the customers waiting for too long. This Markovian approach has its own drawbacks though. For one, this type of analysis usually requires assumptions on the statistics of the parameters that need not be verified in practice. In our example, the arrival of orders need to follow a Poisson process. Further, even with these assumptions, explicit formulas are not always forthcoming and numerical solutions methods have to be called upon which limits the applicability of the approach. Another drawback is that the computational complexity grows with the size of the networks, and exact analysis become computationally prohibitive. This curse of dimensionality also occurs in solving MDPs.

A part of my research is devoted to obtaining analytical insights from stochastic models that are otherwise difficult to solve exactly. This line of research falls within the widely studied field of what can be called asymptotic analysis. The main idea here is to investigate the model for a set of parameters in a regime that is amenable to analysis but still has some relationship to the parameters of interest. To illustrate this, in our manufacturing facility, if too much capacity is installed compared to the demand, then machines will remain idle most of the time and the investment will not be profitable. On the other hand, if the capacity is too low then there will be a backlog. There is usually a sweet-spot that balances these two objectives. To determine this spot, one takes the size of the network to infinity. It so happens that, in this asymptotic regime, the analysis simplifies since the dynamics of the system become

more and more regular. One then computes the capacity to be installed using this simplification and provides guidelines to be used in practice when the system-size is finite. Asymptotic analysis thus can be applied to obtain easy-to-compute approximations and bounds that can then be used in lieu of the original objective function in the optimization problems stated above. Such techniques are as old as probability theory itself with Gaussian approximation and Brownian motion being two celebrated examples. Of course, as new applications emerge, these techniques have to refined, improved and adapted to these new environments.

Asymptotic techniques in stochastic networks are quite often inherited and inspired from those developed in other fields such as physics in which stochastic models are also widely used. These include fluid or hydrodynamic limits [5], diffusion limits [6], large deviations [7] and mean-field approximations [8]. Intuitively, fluid limits are useful when the number of objects present in the network is so big that, at a macroscopic level, the stochastic dynamics becomes less preponderant. At this scale, the movement of the objects through the network starts to resemble that of a fluid. The dynamics can then be described by differential equations driven by the average drift. Diffusion models are a refinement of the fluid model in which the deviations from the fluid trajectories are studied on a scale at which stochasticity is not completely washed out. It is similar to the Gaussian refinement for the law of large numbers but for stochastic processes that evolve over time. Large deviations looks at the probability of trajectories moving even further away from those prescribed by that of the fluid limit. In the mean-field approximation, the size of the network itself grows and one looks at the fraction of nodes that find themselves in a certain state. Again, in the limit, the stochastic component is washed out and the interaction between the nodes can be summarized by averages. Differential equations can again used to described how these fractions evolve. These techniques (as well as some others) have been widely used to obtain approximations which can then be plugged into the optimization and control subroutines [9, 10, 11]. In the first part of the manuscript, three applications of some of these techniques will be presented.

The second part of the manuscript is on the distributed analog of the above setting. Some of the networks considered are typically shared by different entities with conflicting objectives. For example, the communication network in a country may

be shared by the several operators providing Internet services. Each operator will want to maximize the data rate of its customers in a selfish way and will not be concerned with the influence of its decision on the performance of the other operators. Since the capacity of the network is limited, the improvements seen by one operator is usually at the expense of the others. Such conflictual situations are modelled by non-cooperative game theory [12]. It predicts what strategy entities or players will employ in such situations. Since there is no single objective function that suits all the entities, the notion of optimal solution is replaced in game theory by that of a Nash equilibrium. It is an equilibrium in the sense that no entity has an incentive to unilaterally deviate as long as the others follow the equilibrium strategy. There is a vast literature on different and interesting applications of game theory in communication networks as well as in other domains [13, 14, 15].

One of the consequences of distributed or selfish decision-making is that the equilibrium reached may be inefficient from an social point of view. As an illustration, consider a road-traffic network where each vehicle wants to take the shortest route to its destination. Since each vehicle is selfishly optimizing its travel time, the routing choices are determined by an equilibrium. The average travel-time over all the vehicles at the equilibrium will surely be no better that if the routes were decided by a single entity with that objective in mind. Starting from this observation, one line of study seeks to know how bad can the consequences of selfish choices be. This is usually measured by *Price of Anarchy* (PoA) which is ratio of the worst-case performance at the equilibrium to the optimal social cost [15]. One of the goals here is to see how much performance is being lost due to a distributed (or decentralized) architecture which may be either inherent or willfully chosen. There are situations in which it is not practically feasible to have a centralized control (road-traffic, Internet, etc.). In some other contexts, distributed architectures are preferred due to their robustness to failures as well as scalability. Here each controller takes decisions based on its local information which may be different from that of the others. Even though they may not have selfish motives, the asymmetry in information can also sometimes lead to an equilibrium that is different from the desired optimum. In both contexts, if the PoA is high, some measures may be needed to reduce it. This can be done by, for instance, tolls or taxes in the first case and by more exchange of

information in the second. A low PoA on the other had means that the system can be operated in a distributed fashion without much loss in performance. Of course, this is a worst-case analysis and suffers from similar drawbacks that were seen in the approach of approximation algorithms, *i.e.*, the worst-case scenario may be too pessimistic and may not occur in practice. Nevertheless, it does give a metric to compare the two architectures. For the purposes of this manuscript, the term asymptotic analysis is used in a broader sense to include worst-case analysis (computation of PoA) in games. Two applications of PoA in communication networks will be presented in this part.

Organization of the manuscript

The manuscript is divided into three parts with the first two devoted to previous research and the third one to possible new directions. The first part consists of three chapters and the second one has two. Each chapter summarizes the contributions and the results from one paper.

Part I focuses on centralized systems and asymptotic analysis in the spirit of the second approach of stochastic models. Chapter 1 considers load-balancing with finite buffers. From the asymptotic analysis, a dimensioning rule is obtained to determine the number of servers as a function of the target blocking probability and the arrival rate of the demand. In Chapter 2, the problem of traffic surges in bandwidth-sharing networks is studied. In order to protect other non-surging flows, the surging ones are penalized by reducing appropriately the weights that determine their share of the node capacities. Chapter 3 investigates a scenario in which the control policy cannot be applied at arbitrary instants. It looks at what can happen if the control instants do not happen sufficiently often enough compared to the evolution of the underlying system.

The asymptotic analysis in Part II is in the spirit of the first approach of worst-case analysis for two routing problems in distributed systems. For the single-path routing problem, Chapter 4 presents an approximation algorithm based on the best-response algorithm in game theory. It is an example of a distributed algorithm for solving a centralized optimization problem. In the last chapter (Chapter 5), worst-

case analysis is carried out for a purely distributed and non-cooperative load-balancing problem but this time for splittable flows.

PART I

Centralized systems

CHAPTER 1

LOAD-BALANCING IN BLOCKING SYSTEMS

Consider a single dispatcher which receives tasks that have to be routed immediately to one of several servers. Each server has speed of 1 and maximum buffer length of θ. Tasks with independent and identically distributed (i.i.d.) durations with mean 1 arrive at the dispatcher according to a Poisson process of rate λ. The well-known Erlang-B or the $M/M/n/n$ queue is a particular case of this model obtained when θ is set to 1 and the service-times are exponentially distributed.

Given the current number of tasks at each server, the dispatcher must allocate the incoming task to one of the servers with the objective of minimizing the overall blocking probability. The optimal policy depends upon several parameters among which the most studied ones are the arrival process, the service-time distribution, and the service discipline at each server.

The intuitive policy of sending to the shortest queue, join-the-shortest-queue (JSQ), is optimal under certain conditions. Using dynamic programming, [16] proved the optimality of JSQ for general interarrival processes and exponential service times, while [17] used stochastic ordering techniques to extend this result to state-dependent service-times. In fact, JSQ is also optimal for other objective functions, notably for mean sojourn time when $\theta = \infty$. In this case, the sufficient conditions for optimality are less restrictive [18]. On the other hand, counterexamples are known that show that JSQ is not optimal for certain types of service-time distributions [19].

These optimality results were derived assuming the first-come-first-serve (FCFS) service discipline at the servers. For other disciplines such as processor-sharing (PS), which is employed in computer systems, or for service-time distributions that do not fall within the stipulated class, optimality results are not simple to obtain. Furthermore, even for the easy case of FCFS and exponentially distributed service-times,

there is no simple expression for the stationary distribution of the state-vector and, by consequence, for the blocking probability as a function of the θ or n. Approximations [20] or asymptotic analysis [21] have been called upon to determine how the blocking probability depends on these parameters.

Contrast this with the Erlang-B queue for which the stationary distribution has a simple product-form as long as the arrivals follow a Poisson process and task-sizes are independent. Moreover, this formula depends on the service-time distribution only through its mean. That is, the knowledge of the mean service-time is sufficient to compute the steady-state performance measures irrespective of the other statistics of the service-time distribution. Thanks to the existence of a formula, the following simple dimensioning rule for systems with high arrival rates was proposed in [22]:

$$n = \lambda + \beta\sqrt{\lambda}, \tag{1.1}$$

for some $\beta \in \mathbb{R}$, for efficient use of the servers. Here, efficiency means that the servers are utilized at close to their maximum capacity while at the same time the probability of blocking is close to 0. This is a thin line on either side of which one or the other of these two quantities will be unsatisfactory. For example, if n is larger than the prescribed scaling, then a fraction of servers will always be free meaning that capacity is wasted. On the other hand, if n is smaller, then the probability of blocking will be strictly positive leading to unsatisfied customers. A similar scaling law was later derived for the $M/M/n$ queue (i.e., queue without blocking) by [23] and is known as the *Quality and Efficiency Driven* (QED) regime. Such conclusions are not straightforward to derive when the stationary distribution is not available as is the case with JSQ.

A natural question is then to ask whether there exist scenarios in which similar formulas can be obtained for values of $\theta > 1$. The product-form formula of the Erlang-B queue is a consequence of the reversibility of the Markov chain describing the number of tasks in the system. Another consequence of reversibility is insensitivity to the service-time distribution. That is, both these desirable properties can be deduced by checking for reversibility of the underlying Markov chain.

It turns out that reversibility can be shown for more general stochastic networks under certain conditions [24, 25]. Main among those are that the load-balancing pol-

icy has to have a balance property, and the service discipline has to be a symmetric one. Unfortunately, JSQ does not satisfy the balance property and FCFS is not symmetric. Nevertheless, there are other disciplines including PS, that are symmetric, and which happen to be used in computer systems. In such cases, reversibility could be aimed for with appropriately choosing the routing policies. Within the class of reversible policies, it was shown in [26] that the optimal one routes to server i with probability:

$$\frac{\theta - x_i}{\sum_{j=1}^{n}(\theta - x_j)} 1_{x+e_i \in \mathscr{X}}, \tag{1.2}$$

where x_j is the number of tasks in server j, x is the vector of the state and $\mathscr{X}$ is the set of feasible states. Further, there is a product-form formula for this policy that generalizes the one for $M/M/n/n$ queue.

In summary, although the insensitive policy in (1.2) is not globally optimal, it has the following properties: *(i)* it has a closed-form formula for the stationary distribution; *(ii)* formula is insensitive to the service-time distribution and only requires Poisson arrivals; *(iii)* it is optimal in the class of insensitive policies and permits analysis of disciplines such as PS.

Contributions

In [JP18], we subject the optimal insensitive policy to different types of asymptotic analysis when the number of processors n gets large. We first obtain the stationary measure of the number of occupied servers and give its transient mean-field limit. Considering the symmetric version of the model, we show that the functional law of large numbers also holds for the stationary version of the system (limits in n and t commute). The existence and uniqueness of the limiting stationary probabilities are proved through a monotonicity argument involving the Erlang formula, while the stationary point is characterized through the Erlang formula. This implies simple conclusions on the asymptotic behavior of the blocking probability: the blocking probability is asymptotically vanishing for the sub-critical ($\rho < 1$) case and is equal to $1 - \rho^{-1}$ for the super-critical case $\rho > 1$. In both cases, this blocking probability corresponds to the optimal blocking probability achievable by any non anticipating policy. Of course this is far from being sufficiently informative and we

are led to focus on a more detailed study of the stationary distribution for large n, establishing both large deviations principles for sub- and super-critical cases and moderate deviations results. We show that, when $\rho < 1$ is fixed, the blocking probability is exponentially small, and we characterize the most probable deviations from the mean-field limit. The large deviation cost is shown to be a sum of two terms: the "distance" to the stationary point from distributions with a given mean plus the cost of having a different mean from the true stationary mean. We also show that a central limit theorem is valid for the occupation numbers around the stationary point of the mean-field in the sub-critical regime. For the critical case $\rho = 1$, the right scaling is not any more of order $\sqrt{n}$. Using local limit theorems and exploiting the characterization of deviations from the mean-field limits, we show that the number of free servers scales like $n^{\frac{\theta}{\theta+1}}$, the limiting distribution depending on θ and coinciding with the normal distribution only for $\theta = 1$. In a third step, we study the critical case at a finer scale and show that a qualitative phase transitions occurs at the critical load $\rho_c(n) = 1 - an^{-\frac{\theta}{\theta+1}}$ where θ is the buffer depth. The blocking probability is exponentially small until $\rho_c(n)$ and of order $n^{-\frac{\theta}{\theta+1}}$ at this critical load. This generalizes the Halfin-Whitt regime established for the $M/M/n/n$ system, and shows that the popular staffing rule established for the $M/M/n/n$ system does actually change with the value of θ when load-balancing is employed. The super-critical regime is simpler to characterize, the deviations being of order 1. We illustrate these findings on simple numerical experiments.

1.1 Mean-field limit and deviations

In the analysis of queueing systems, the state is typically taken to be the number of tasks in each server. When $n \to \infty$, it is more informative to work with the number of servers containing a certain number of customers. Let $\{S^{(n)}(t) \in \mathscr{S}\}_{t\geq 0}$ be a stochastic process denoting, at time t, the number of servers with i tasks, $i = 0,\ldots,\theta$. Under Poisson arrivals and exponentially distributed job sizes, $S^{(n)}(t)$ is a continuous-time

jump Markov process on the state space $\mathscr{S}$ with the following transition rates

$$S^{(n)}(t) \to \begin{cases} S^{(n)}(t) + e_i - e_{i-1} & \text{at rate } \lambda_{i-1}(s), i \geq 1; \\ S^{(n)}(t) + e_i - e_{i+1} & \text{at rate } s_{i+1}, \end{cases} \tag{1.3}$$

assuming that the transitions take the process to a state within $\mathscr{S}$.

Theorem 1. *If the job-size distribution is exponential, the process $S^{(n)}(t)$ is a reversible Markov process and its stationary distribution is given by*

$$\pi^{(n)}(s) = \pi_0^{(n)} \frac{(n\theta - \bar{s})!}{(n\theta)!} \binom{n}{s} \prod_{k=0}^{\theta} \left(\frac{\theta!}{(\theta-k)!} (n\rho)^k \right)^{s_k}, \tag{1.4}$$

where $\bar{s}$ is the total number of jobs in the system, and $\rho = \lambda/n$ is the load per server, and $\pi_0^{(n)}$ corresponds to the probability of the state with all servers empty, that is, $\bar{s} = 0$ and $s = (n, 0, \ldots, 0)$.

Corollary 1. *Using the PASTA property, the blocking probability is given by*

$$B_\theta^{(n)} = \pi_0^{(n)} \frac{(n\rho)^{n\theta} (\theta!)^n}{(n\theta)!}. \tag{1.5}$$

The mean-field limit of (1.3) is obtained for a fixed ρ when $n \to \infty$. It describes the evolution of the server configurations over time starting from a given initial configuration. For technical ease, it will be assumed that the service-time distribution is exponential though we expect similar limits to hold for arbitrary distributions. Without the exponential assumption, the proof becomes much more technical as one has to work with measure-valued processes. Such results have been proved though for policies such as join-the-shortest-of-d queues with generic service-time distribution in [27], which makes us hopeful.

Theorem 2. *Fix a $\rho < 1$ and a $\theta \geq 1$. For exponentially distributed job-sizes, for all fixed time, $S^{(n)}(t)/n \to y(t)$, in probability, with y which is the solution of the following set of differential*

equations:

$$\frac{dy_j(t)}{dt} = \rho\frac{\theta-(j-1)}{\theta-\sum_k k y_k(t)} y_{j-1}(t) + y_{j+1}(t) \tag{1.6}$$

$$-\rho\frac{\theta-j}{\theta-\sum_k k y_k(t)} y_j(t) - y_j(t),\ 0<j<\theta, \tag{1.7}$$

$$\frac{dy_\theta(t)}{dt} = \rho\frac{1}{\theta-\sum_k k y_k(t)} y_{\theta-1}(t) - y_\theta(t), \tag{1.8}$$

$$\frac{dy_0(t)}{dt} = y_1(t) - \rho\frac{\theta}{\theta-\sum_k k y_k(t)} y_0(t). \tag{1.9}$$

with $y(0) = \lim_{n\to\infty}\frac{S^{(n)}(0)}{n}$.

Theorem 3. *For* $0<\rho<1$*, the unique steady-state solution of the system of equations* (1.6)–(1.9) *is given by*

$$\hat{p}_j = \left(\frac{\theta-\hat{c}}{\rho}\right)^{\theta-j}\frac{1}{(\theta-j)!}\hat{p}_\theta, \tag{1.10}$$

$$\text{with } \hat{p}_\theta = \frac{1}{\sum_{k=0}^{\theta}\left(\frac{\theta-\hat{c}}{\rho}\right)^k\frac{1}{k!}}. \tag{1.11}$$

where

$$\hat{c} = \theta - \rho\zeta_\theta^{-1}(1-\rho), \tag{1.12}$$

with ζ_θ^{-1} *as the inverse function of the Erlang blocking viewed as a function of the traffic intensity for a fixed buffer depth* θ.

If $\rho>1$*, the unique solution is* $\hat{c}=\theta$*,* $\hat{p}_j=0$*, for* $j\le\theta-1$ *and* $\hat{p}_\theta=1$.

Theorem 3 gives the fraction of servers that will be have a certain number of tasks in the mean-field limit. Since in practice, the limit is never attained, one can study how far we deviate from it for different values of ρ as $n\to\infty$. Of particular interest is the critical case $\rho=1$. For $a,z\in\mathbb{R}$ and $\theta\ge1$, define

$$\widehat{\Phi}_\theta(z;a) = \int_z^\infty \exp\left(au - \frac{u^{(\theta+1)}}{(\theta+1)!}\right)du. \tag{1.13}$$

For $\theta=1$ and $a=0$, $(2\pi)^{-1/2}\widehat{\Phi}_\theta$ reduces to the complementary cumulative distribution

function of the standard normal distribution.

Theorem 4. *For $\rho = 1$ and $z \in \mathbb{R}_+$,*

$$\lim_{n\to\infty} \mathbb{P}\left(\frac{S^{(n)}_{\theta-1}(\infty)}{n^{\theta/(\theta+1)}} > z \right) = \frac{\widehat{\Phi}_\theta(z;0)}{\widehat{\Phi}_\theta(0;0)}, \tag{1.14}$$

Corollary 2. *For $\rho = 1$, $\theta = 1$, and $z > 0$,*

$$\lim_{n\to\infty} \mathbb{P}\left(\frac{S^{(n)}(\infty)}{\sqrt{n}} > z \right) = 2(1 - \Phi(z)), \tag{1.15}$$

where Φ is the distribution function of the standard normal distribution.

That is, for $\rho = 1$, conditioned on being accepted, a customer has a high probability of being routed to a server $\theta - 1$ jobs. This property has a direct consequence on the state information the dispatcher needs to take routing decisions. We shall elaborate upon later on.

1.2 Blocking probability in the Halfin-Whitt regime

In the mean-field limit, it can be shown that the insensitive load-balancing policy achieves the lowest possible blocking probability within the class of non-anticipative policies independently of θ.

Proposition 1. *The limiting blocking probability of the insensitive load balancing policy is given by*

$$B_\theta = \begin{cases} 0 & \text{if } \rho < 1; \\ 1 - \rho^{-1} & \text{otherwise.} \end{cases} \tag{1.16}$$

Since the blocking probability is independent of θ, even a buffer of size 1 is sufficient to get the optimal stationary behavior. This optimality is valid only in the limit $n \to \infty$. In order to compute the blocking probability (or other performance measures) for values of n that are large but finite, one has to look at finer scales.

For $\rho < 1$, it can be shown that the blocking probability goes to 0 exponentially quickly in n while for $\rho > 1$ it goes to $1 - \rho^{-1}$, a strictly positive quantity. We next

look at the interesting case when $\rho \to 1$ together with $n \to \infty$. It will be shown that the blocking probability vanishes at a polynomially rate in this regime and that this polynomial depends on the buffer size θ.

For the Erlang C model, that is, a system without blocking, this is the QED regime and it has the following interpretation: if the cost of servers is high, Halfin and Whitt [23] observed that it could be beneficial to reduce the number of servers in such a way such that the probability of waiting is strictly positive and less than 1 instead of going to 0 with the number of servers. This increase in the waiting probability has the benefit of requiring $\lambda + O(\lambda^{1/2})$ instead of $b\lambda$, $b > 1$, servers. Thus, one gains in the cost and the utilization of servers at the expense of the waiting probability. The name QED evokes the trade-off between these two quantities. We note that this asymptotic regime was already studied for the Erlang B system in Jagerman [22] (see Theorem 14) but the interpretation in terms of a trade-off is due to Halfin and Whitt [23] for systems without blocking and Whitt [28] for systems with blocking. These works were followed by more precise asymptotics in both the Erlang B as well as the Erlang C systems [29, 30].

In addition to the classical asymptotic regimes, such as mean-field and QED, one can define an intermediate regime known as the Non-Degenerate Slowdown (NDS) regime (see [31]). The feature of the NDS regime is that the mean waiting time is of the same (non-degenerate) order as the mean service time. For the insensitive load-balancing policy that we are investigating, this relationship between the mean waiting time and the mean service time is verified irrespective of the load because the buffer length is finite and there is a non-zero probability of routing an arrival to a non-empty queue. Thus, the NDS regime is rather trivial for this policy, and is not investigated here.

The following theorem gives the QED scaling for the balanced load-balancing policy, and can be viewed as a generalization of the QED result for the Erlang loss model.

Theorem 5. *For $a \in (-\infty, \infty)$, let*

$$n\rho = n + an^{1/(\theta+1)}. \tag{1.17}$$

Then,

$$\lim_{n\to\infty} B_\theta^{(n)} n^{\theta/(\theta+1)} \int_0^\infty \exp\left(au - \frac{u^{(\theta+1)}}{(\theta+1)!}\right) du = 1. \tag{1.18}$$

Note that a can be positive or negative, which means that even with a total load larger than the number of servers, the blocking probability can decay to 0 provided that (1.17) is satisfied asymptotically. For $a = 0$, using simple computations, Theorem 5 leads to the following corollary:

Corollary 3. *If* $\rho = 1$*:*

$$B_\theta^{(n)} \sim \frac{(\theta+1)!^{\frac{1}{\theta+1}}}{\theta+1} \Gamma\Big(\frac{1}{\theta+1}\Big) n^{-\theta/(\theta+1)}, \tag{1.19}$$

where Γ *is the Gamma function.*

Note that for $\theta = 1$, we retrieve that:

$$B_1^{(n)} \sim (0.5\pi n)^{-1/2}. \tag{1.20}$$

Remark 1. *It turns out that the scaling of 1.17 is the same as that in [32] (see display* (4.9)*), and differs only in the context in which it is obtained. The one in [32], which they call the QED-c regime, appears in the context of an Erlang C system with abandonments in which the arrival rate has an additive uncertainity. The parameter c, which in our model translates to* $1/(\theta+1)$*, is the exponent of the additive uncertainity in [32].*

1.3 Performance planning and practical schemes

The asymptotic analysis of insensitive load-balancing gives a conservative planning tool for managing the performance relationship between the load ρ and the delay guarantees depending on θ, and the blocking guarantees depending both on n and θ. Indeed, in many applications, a given level of quality of service in terms of delay has to be reached and this can be done by fixing θ. For a given buffer depth θ the mean delay of a job entering the system will be less than θ (the server speed and the mean job-size are fixed to 1). On the other hand, for a given θ and n, we have precisely characterized the asymptotics of the blocking probability, unveiling

the critical load $\rho_c(n)$ as the frontier of the acceptable blocking probability for most applications. Hence, one can adapt the number of servers n to cope with a target blocking probability given the load or adapt the load given the number of servers. Note that this planning is not straightforward for specific sensitive policies.

Another second interpretation is by considering the staffing rule which is the number of servers necessary to obtain a vanishing blocking probability in the limit when the total charge is large. In [22] and [28], the staffing rule for $\theta = 1$ was shown to be $\lambda + O(\lambda^{1/2})$, that is, at least these many servers are required to get a vanishing blocking probability when λ is large.

Theorem 5 generalizes the known results for $\theta = 1$ to larger values of θ leading to the following staffing rule:

Proposition 2. *For a fixed target blocking probability, the number of servers should scale as $\lambda + a\lambda^{1/(\theta+1)}$, where a is determined by the target blocking probability and can computed using (1.18).*

One of the major criticisms of state-dependent policies such as JSQ or the optimal insensitive policy is that the dispatcher needs to know the state of every server in order to route an incoming job. The process of collecting state information can add significant delays and lead to lost revenue [33]. Practical policies such as the JSQ(d) [34] or the JIQ [33] play on the trade-off between information and optimality, and aim to perform much better than state-independent policies while at the same time needing much less information than the whole set of servers. For example, JSQ(d), with the knowledge of the state of only d (which can be fixed number independent of n) servers, has a considerable gain at least in the case of exponentially distributed job-sizes and in the absence of blocking when $d = 2$ compared to $d = 1$.

While at first glance the insensitive load-balancing policy seems to require full state information, in heavy-traffic, Theorem 4 can be of help. It has the following implication: for $\rho = 1$ and n large, most of the servers will have either θ or $\theta - 1$ jobs. One possible scheme to exploit this property is based on the idea first proposed for JIQ, which was motivated by the observation that collecting state information at arrival instants should be avoided in order to reduce delays for jobs. In JIQ, the servers inform the dispatcher (or leave information on a bulletin board) when they become

idle. The dispatcher then knows which servers are idle, and it routes an incoming packet to one of these servers, if there is one, otherwise it routes based on no information. Thus, upon arrival a job can be routed immediately based on state information collected previously.

For the insensitive policy, one can conceive a scheme in which servers inform the dispatcher whether they have $\theta-1$ or fewer than $\theta-1$ jobs (this scheme automatically implies that the dispatcher also knows which servers have θ jobs). When a job arrives, the dispatcher will need to determine the state of only those servers with less than $\theta-1$ jobs. Since this number is expected to be on a smaller scale than $n^{\theta/(\theta+1)}$ (thanks to Theorem 4), one can expect to reduce the information flow between the servers and dispatchers at arrival instants. A back of the envelope calculation based upon the proof of Theorem 4 leads one to believe that there will $O(n^{(k+1)/(\theta+1)})$ servers with k jobs and hence $O(n^{(\theta-1)/(\theta+1)})$ servers with less than $\theta-1$ jobs but this remains to be rigorously investigated.

CHAPTER 2

TRAFFIC SURGES IN BANDWIDTH-SHARING NETWORKS

We now turn to the more general model of flows traversing a network and sharing the bandwidth of the links that they have in common. The load-balancing problem treated in the previous chapter can be seen as a special case of a network with parallel links between the source and the destination. The stochastic dynamics of these flows, which depends upon on how the bandwidth of the links are allocated to the flows, can be modelled by *bandwidth-sharing networks* [35, 36, 37] which have become a standard modelling tool in communication networks. In particular, they have been used extensively to represent the flow level dynamics of data traffic in wired or wireless networks [38], as well as for the integration of voice and data traffic [39], hence generalizing more traditional voice traffic models, e.g. [24]. A more recent overview of these network is presented in [40].

As in the previous chapter, the analysis of the usual performance measures such as the average delay of the flows in the network is a complex task and asymptotic analysis (fluid-limits, heavy-traffic) is typically done to get insights [41, 37]. The scaling limits in these works is obtained by scaling all the flows simultaneously. Although this type of joint-scaling is useful in a variety of situations, there are others in which only a subset of classes undergoes an abnormal traffic surge while the other classes remain on the usual scale. One example of this phenomenon is *s*lash-dot-crowds or flash-crowds on web servers and content distribution networks [42]. They can occur due to either some unexpected event (catastrophies, *e.g.*) or due to a highly anticipated event (sporting finals, *e.g.*) for which the network has not been properly dimensioned. Another reason for these surges can be malicious denial-of-service-attacks [43].

To protect content providers from such surges, several mechanisms have been

propsoed [44, 45, 42]. However, in addition to overloading the content providers, a traffic surge can also negatively impact the performance of other concurrent flows in the network. The temporarily unstable class can potentially starve the other classes from network capacity thereby subjecting them to unreasonable delays and packet losses. In such circumstances, in addition to protection mechanisms in web servers, it is also important to implement bandwidth-sharing mechanisms inside the network that would protect the stable classes from the adversarial effects of the surge. It then seems natural that such mechanisms should penalize the temporarily unstable class more when the level of congestion it creates is larger, without actually throttling it. Thus, the more significant the surge due to a certain class is, the smaller the bandwidth each flow in this class gets. To the best of our knowledge, the consequences of traffic surges on the performance of the different classes in the presence of such bandwidth-sharing mechanisms have not been explored.

In this chapter, we present an analytic treatment of the interactions that takes place between the temporarily unstable classes (or classes that undergo a surge) and the stable classes during a traffic surge when the temporarily unstable class is penalized proportionally to its level of congestion.

Integration of streaming and elastic traffic

As an example, to illustrate this interaction and how asymptotic analysis can be helpful, we shall consider a network where two intrinsically different types of traffic – "streaming" and "elastic" – share a link of capacity 1. Such models have been considered by [46, 39]. The streaming traffic is assumed to require a fixed rate, say, c per flow. On the other hand, the elastic traffic, as the name suggests, doen not have a strict requirement and is satisfied with a time-varying rate.

When there are x_1 (resp. x_2) flows of elastic (resp. streaming) traffic, the rates are allocated as follows:

$$\phi_1(x) = \min\left(\frac{r_1 x_1}{r_1 x_1 + c x_2}, 1 - c x_2\right),$$
$$\phi_2(x) = c x_2,$$

where the parameter r_1 quantifies the level of priority. The allocated capacity to class 1 cannot exceed its fair share, which is assumed to be $r_1 x_1/(r_1 x_1 + c x_2)$. Since each streaming session is guaranteed a rate c, the total allocation to class 1 cannot exceed $1 - c x_2$ either. The allocated capacity cannot exceed the total capacity, and the state-space must be restricted to states x_2 such that $\phi_1(x) + \phi_2(x) \leq 1$. Then, if the number of current streaming flows x_2 is such that $\phi_1(x + e_2) + \phi_2(x + e_2) > 1$, arriving streaming flows must be blocked from the network. The capacity that remains is a provision for future incoming streaming calls.

In [47], fluid and diffusion approximations are derived for a similar model when both streaming and elastic flows are evolving at a fast time-scale. In contrast, in this chapter we will be interested in the blocking probability of streaming flows when the elastic flows undergo a traffic surge which means that only the elastic flows will be scaled to a deterministic limit whereas the streaming flows retain their stochastic nature. When streaming and elastic traffic do not interact, the probability of blocking can be computed using an Erlang Fixed-Point approximation [48]. However, in the context of the present example, the interaction of these two types of traffic makes it more difficult to apply these fixed-point approximations, mainly due to the fact that the state space of the elastic flows is unbounded. In the regime when r_1 is small, we propose a rule-of-thumb, based on our result in Theorem 6, that can guarantee a blocking probability smaller than a desired value.

Contributions

In [FJP14], we introduce a scaling where the initial conditions of a possibly only a subset of classes (the surging classes) converges to infinity. To handle the surge, the network will penalize this class by reducing its priority accordingly. In the example above, the parameter r_1 will be scaled inversely to the size of the surge. Such a scheme ensures that the stable classes are protected from the surge, while at the same time the surging classes are not completely shut out.

In order to obtain a classical fluid limit for Jackson networks [49] or for more complex bandwidth-sharing networks [37], all the classes are jointly scaled in time and in space. This yields a set of differential equations that govern the dynamics of all the classes. Under additional assumptions on the drift δ of the considered Markov

process, the differential equation is simply of the form $\dot{x}(t) = \delta(x(t))$ (see the considerable amount of work on fluid limits and ODE methods both for Markov processes and for communications networks [49, 50, 51, 37, 52]).

In our case, the situation differs as the transitions of surging classes are also scaled to model that the priority weight of surging classes is inversely proportional to the level of surge. This has some impact on the the structure of the limiting process. Under this scaling, we will show that the dynamics of the temporarily unstable classes can be described by a deterministic differential equation, while the stable classes retain their stochastic nature. Hence, a time-scale separation occurs: the temporarily unstable classes evolve on a much slower time-scale compared to the stable classes. However even with this separation of time-scales, a strong coupling in the dynamics of the temporarily unstable and the stable classes remains. The dynamics of the temporarily unstable class is influenced by the stable classes through their conditional stationary distribution (conditional on the level of congestion of surging classes flows being fixed to its present macroscopic value), which in turn depends on the temporarily unstable classes. Hence, for surging classes the differential equation obtained is of the form $\dot{x}(t) = \bar{\delta}^t(x(t))$, where $\bar{\delta}^t$ is an average of the first coordinate drift according to the conditional distribution of the other classes, given the state of the surging classes.

Our first contribution is in establishing the convergence in L^1 uniformly on compact sets in bandwidth-sharing networks with a mix of surging classes and stable classes. Second, we introduce the notion of robust stability, which describes a situation when the network can absorb a surge of traffic by:

— keeping the non-surging classes stable,

— reducing the macroscopic state of surging classes to 0.

We call the set of traffic parameters that lead to these two conditions, the robust stability region. We characterize the robust stability region for work conserving allocations and for monotone allocations. We first show that for work conserving allocations, the unstable classes, at their macroscopic time scale, see the other classes as having full priority, while the effect of the surging classes on the other classes gradually vanishes (again at a macroscopic time scale). Hence surging classes tend macroscopically to 0 under the (usual) stability condition of the system $\sum_{j=1}^{N} \rho_j < 1$. The

situation is more complex for non work-conserving networks, where the behavior of the surging classes depends in an intricate manner upon that of the other classes. In particular, under the usual stability conditions of the network, the macroscopic state of surging classes might converge to 0 or to a strictly positive number, depending on the conditional distribution of the other classes. We prove for monotone networks that only when the allocation giving full priority to the stable classes, that is the allocation in a network where surging classes have 0 arrival rate, is stable, then surging classes converge to 0 on the macroscopic time scale. This hence demonstrates that the robust stability region might be strictly included in the (usual) stability region.

2.1 Asymptotic analysis of surges

Consider a bandwidth-sharing network with N traffic classes. Within each of the N traffic classes, resources are shared according to a processor-sharing service discipline. Class-i customers arrive according to a Poisson process of intensity λ_i and require exponentially distributed service times of mean μ_i^{-1} for class-i so that the load of class i is $\rho_i = \lambda_1 \mu_i^{-1}$. The arrival processes of all classes are mutually independent. Our main results allow for time-varying arrival rates for the class exhibiting a traffic surge. When applicable, we reflect this dependence in the notation by adding the time parameter to the arrival rates and then $\lambda_1(t)$ is the arrival rate of class 1 at time t.

Let $X_i(t)$ be the number of flows in the network at t and let $S \leq N$ denote the number of classes that undergo a surge. The weights r_i of the surging classes are chosen inversely proportional to the size of the total initial surge, x_0, *i.e.*,

$$r_i = r_i(|x_0|) = \frac{\omega_i}{|x_0|},$$

for some $\omega_i \geq 0$.

Set $K = |x_0|$ and let Y^K denote the (scaled) process:

$$Y^K(t) = \left(\left(\frac{X_i(Kt)}{K} \right)_{i=1\ldots S}, (X_i(Kt))_{i=S+1\ldots N} \right). \tag{2.1}$$

Here, the surging class are scaled in both time and space where as the stable classes are only scaled in time.

Our main result formalizes the intuition mentioned in the previous section that as the surge grows to infinity (*i.e.* $K \to \infty$), the scaled stochastic process Y^K converges to one whose first S coordinates are deterministic and which are a solution of a differential equation that can be described in terms of an averaged rate $\bar{\phi}$. For the other $N-S$ coordinates, corresponding to the stable classes, the dynamics remains stochastic. However, the transient dynamics of these classes is washed out on this macroscopic time-scale and only the stationary behaviour is observed. In the limit, the result implies a time-scale separation between the surging classes and the other ones.

Define U^z to be a $N-S$ dimensional Markov birth-and-death process with arrival rates λ_i and death rates $\phi_i(z,\cdot)$, $i = S+1,\dots,N$, where $z \in \mathbb{R}_+^S$. This the dynamics of the stable classes for a fixed value z of the number of flows of the surging classes. Denote by $\pi^z(\cdot)$ its stationary probability (when it exists). When we do not use a time index, we implicitly suppose that we consider stationary versions of the processes.

To establish our main result, the following assumptions are needed:

(A_1)**:** $\phi_i(\cdot, x_{S+1},\dots,x_N)$ can be extended to a Lipschitz continuous functions from $\mathbb{R}_+^S \setminus \{0\}$ to $\mathbb{R}_+$.

(A_2)**:** for all fixed z, the process U^z is ergodic. We can thus define $\mathbb{E}U^z$ the mean under the stationary distribution of the process U^z .

(A_3)**:** $\frac{1}{K}\int_0^{Kt} \lambda_i(s)\,\mathrm{d}s \to a_i(t)$, $i = 1\dots S$. It shall be assumed that $a_i(t)$ is differentiable for all t and i.

Although the model was formulated for a constant arrival rate for ease of exposition, the results are proved for time-dependent arrival rates that satisfy assumption (A_3).

Let $u(t) \in \mathbb{R}^S$ be the solution (assuming it exists and it is unique) of the differential equation:

$$\forall\, i = 1\dots S,\ \ \dot{u}_i(t) = \begin{cases} \dot{a}_i(t) - \bar{\phi}_i(u(t)), & \text{if } u_i(t) > 0, \\ 0, & \text{if } u_1(t) = 0, \end{cases} \tag{2.2}$$

with $\bar{\phi}_i(z) = \sum_{y \in \mathbb{Z}_+^{N-S}} \phi_i(z,y)\pi^z(y)$.

Theorem 6. *Under the assumptions* (A_1), (A_2) *and* (A_3), *the process* $Y_i^K(t)_{i=1\dots S}$ *converges in* L^1, *uniformly on compact intervals, to the deterministic trajectory* $u(t)$, *i.e.*

$$\mathbb{E}\left[\sup_{0\le s\le t}\left|Y_i^K(s)-u_i(s)\right|\right]\to 0,\ \ K\to\infty,\ \forall i=1\dots S. \tag{2.3}$$

Moreover, for all times t, *and for all bounded continuous functions* f*:*

$$\lim_{K\to\infty}\mathbb{E}\left(\sup_{0\le s\le t}\left|\int_0^s f\left(Y^K(\theta)\right)\ -\ \mathbb{E}^{U^{Z(\theta)}}\left(f\left(Z(\theta),U^{Z(\theta)}(\theta)\right)\,\middle|\,Z(\theta)=u(\theta)\right)\mathrm{d}\theta\right|\right)\ =\ 0. \tag{2.4}$$

2.2 Qualitative behavior of the limiting processes

The surging classes can exhibit different long-term behaviors depending upon the drift

$$\delta_i(x)=\lambda_i-\mu_i\bar{\phi}_i(x).$$

As $t\to\infty$, it can happen that:

1. the network continues to see the surging classes saturated (at a macroscopic time and space scales), even after any large (macroscopic) amount of time. A sufficient condition is that there exists x such that $\delta_i(x)=0$ and $D\bar{\delta}(x)<0$ and $x>0$, with initial conditions sufficiently close to x. Then the differential equation is asymptotically stable with stable point $x>0$.
2. the differential equation (2.2) governing the dynamics of u is unstable, which means that the traffic surge cannot be absorbed and keeps building up. It might lead to the instability of other classes in the network.
3. the traffic surge will be absorbed at macroscopic time, i.e. the differential equation is asymptotically stable with stable point 0.

A few natural questions arise due to the fact that penalizing the surges can, in some situations, lead to a finite but non-zero amount of fluid remaining in the network. We can ask: under which traffic conditions is the surge completely absorbed? How do commonly analysed allocations such as α-fair [53] or balanced-fair allocation [38] respond to surges?

It is known that all α-fair as well as balanced-fair allocations are stable [54, 36]) if

$$\rho \in \mathscr{S} = \{\eta, A\eta \leq C\}.$$

We shall refer to the interior of the set $\mathscr{S}$ as the "usual conditions of stability". This means that any fluid would drain out eventually if no penalty is imposed on the surges as long as the usual traffic conditions are satisfied. However, when a penalty is imposed, it is interesting to ask how does the penalty influence the dynamics and the stability of these allocations.

To answer these questions, we refine the concept of stability by saying that the network is robust stable if classes undergoing a surge (penalized adequately) eventually drain while the other classes stay stochastically stable. More formally, let $\mathscr{I} \subset [1, N]$ be the set of indexes of surging classes and $\mathbb{P}([1, N])$ the power set of $[1, N]$.

Definition 7. *The network is robust stable if for all $i = 1, \ldots, N$:*

$$\limsup_{t\to\infty} \sup_{\mathscr{I}\in\mathbb{P}([1,N])} \limsup_{|x_j|\to\infty, j\in\mathscr{I}} \frac{E^x[X_i^{|x|}(|x|\,t)]}{|x|} = 0.$$

The robust stability region is the defined as the set of parameters such that the network is robust stable, i.e.:

$$\mathscr{S}^r = \left\{ \rho \in \mathbb{R}_+^N : \ \forall i = 1, \ldots, N, \ \limsup_{t\to\infty} \sup_{\mathscr{I}\in\mathbb{P}([1,N])} \limsup_{|x_j|\to\infty, j\in\mathscr{I}} \frac{E^x[X_i^{|x|}(|x|\,t)]}{|x|} = 0 \right\}. \tag{2.5}$$

Remark that when all classes are undergoing a surge of traffic, we retrieve the usual notion of fluid limit, for which the convergence to 0 implies the network (usual notion of) stability. Hence it holds in general that:

$$\mathscr{S}^r \subset \mathscr{S} = \left\{ \rho \in \mathbb{R}_+^N : \ \forall i = 1, \ldots, N, \ \limsup_{t\to\infty} \limsup_{x_i\to\infty, \forall i=1,\ldots,N} \frac{E^x[X_i^{|x|}(|x|\,t)]}{|x|} = 0 \right\}.$$

We can prove that:

— for work-conserving allocations, the robust stability set coincides with the usual stability set (except possibly on its frontier),

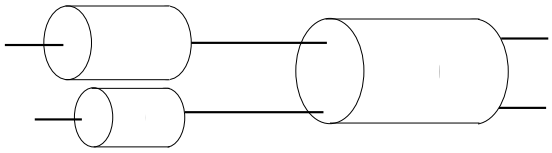


Figure 2.1 – A tree network

— for monotonic networks, the robust stability set coincides with the set of parameters under which a 'priority allocation' is stable and can hence be *strictly included* in $\mathscr{S}$;

The situation is much more complex for non-monotone allocations and a full characterization of the robust stability set is an open problem. The complexity of the dynamics can be explained through the fact that a surging fluid class can "bounce" at 0.

Numerical examples

Example: a tree network Let us consider the tree network shown in Figure 2.1 with $c_1 = 0.4$ and $c_2 = 0.8$. We shall assume the following bandwidth allocation: Define $\mathscr{S}_1 = \{(x_1, x_2) : (r_1 x_1 + r_2 x_2) c_1 < r_1 x_1\}$. For $x_1 > 0$ and $x_2 > 0$,

$$\phi_1(x_1, x_2) = \begin{cases} c_1, & \text{if } (x_1, x_2) \in \mathscr{S}_1, \\ \max\left(\frac{r_1 x_1}{r_1 x_1 + r_2 x_2}, 1 - c_2\right), & \text{if } (x_1, x_2) \in \mathscr{S}_1^c, \end{cases} \tag{2.6}$$

and $\phi_2 = 1 - \phi_1$.

For this network, the allocation becomes a strict priority allocation for class 2 when $r_1 = 0$, in which case class 1 gets capacity c_1 if there are no class 2 flows, and $1 - c_2$ otherwise. Thus, for a fixed value of ρ_2, class 1 is stable if $\rho_1 < \left(1 - \frac{\rho_2}{c_2}\right) c_1 + \frac{\rho_2}{c_2}(1 - c_2)$. The stability regions for $r_1 = 0$ and $r_1 > 0$ are shown in Figure 2.2. This is an example where $\mathscr{S}^r$ is strictly included in $\mathscr{S}$.

The dynamics of $u_1(t)$ for two different values of ρ_1 – one in each region – is plotted in Figure 2.3, for which $\rho_2 = 0.5$. For class 2, when the priority allocation is stable the dynamics of the average number of customers converges to the one of the priority allocation, that is $\rho_2/(c_2 - \rho_2)$, as is illustrated in Figure 2.3.

Figure 2.2 – The different stability regions for the tree network

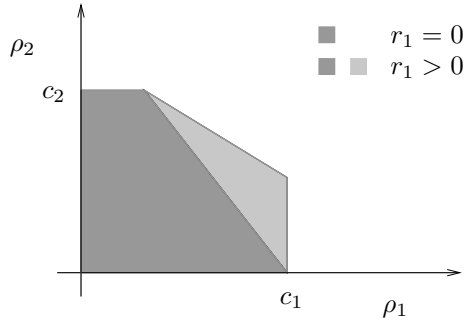
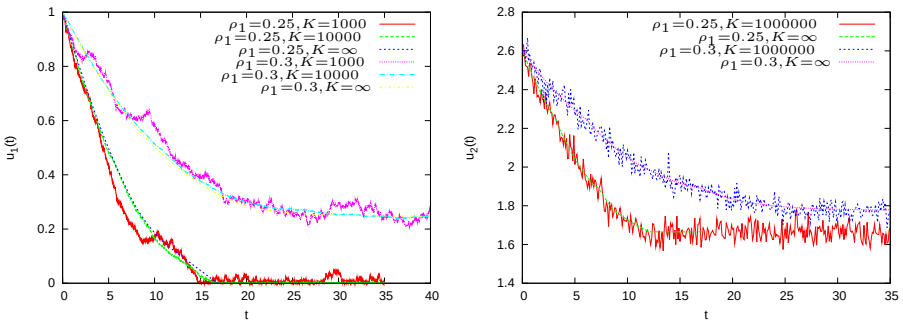


Figure 2.3 – Tree network: scaling of class 1 (left) and of class 2 (right)



Streaming and Elastic traffic We now apply Theorem 6 to the example in which streaming and elastic traffic share the network capacity as explained in beginning of this chapter.

Let p_m denote the desired maximal blocking probability of class-2 flows. We shall set the priority level of class 1 (by varying r_1) such that the probability of blocking of class-2 flows is always less than p_m during the surge. Performing the scaling previously defined, remark that the state-space of class 2 depends, for a fixed macroscopic state z of class 1, on both z and c. Denote

$$\mathscr{S}_z = \left\{ x_2 : \frac{z}{z + cx_2} + cx_2 \leq 1 \right\},$$

the state-space of class 2 given that $u_1(t) = z$. It can be seen that an arrival of class 2 is blocked if and only if there are already $\lfloor \frac{1-z}{c} \rfloor$ calls of class 2 present. Define $\rho_2 = \frac{\lambda_2}{\mu_2 c}$.

The process U_2^z is a birth-death process with birth rate λ_2 and death rate $\mu_2 c x_2$, and whose stationary distribution conditioned on z is given by

$$\pi_2^z(x_2) = \frac{1}{\sum_{j\in\mathscr{S}_z} \rho_2^j/j!} \frac{\rho_2^{x_2}}{x_2!}.$$

For $u_1(t) = z$, the blocking probability of class 2 is then

$$\pi_2^z(x_2)|_{x_2=\lfloor\frac{1-z}{c}\rfloor} =: g(z).$$

Let $\bar{z} := \sup\{z : g(z) \le p_m, 0 < z < 1\}$. Since $g(z)$ is a non-decreasing function of z, in order to guarantee a maximal blocking of p_m, $u_1(t)$ has to be smaller than $\bar{z}$ for all t. This leads us to the following necessary and sufficient condition which guarantees the desired quality of service:

$$\bar{u}_1 := \sup_{0\le t<\infty} u_1(t) < \bar{z}. \tag{2.7}$$

We can compute $\bar{u}_1$ as follows. From Theorem 6, the dynamics of $u_1(t)$ depends on the average capacity allocated to class 1, which an be computed as:

$$\bar{\phi}_1(z) = \sum_{x_2\in\mathscr{S}_z} \min\left(\frac{z}{z+cx_2}, 1-cx_2\right) \frac{\rho^{x_2}}{x_2!} C(z),$$

where $C(z) = (\sum_{x_2\in\mathscr{S}_z} \frac{\rho^{x_2}}{x_2!})^{-1}$. In the case that c is very small ($c \ll 1$), we might consider as a reasonable approximation a Poisson distribution for class 2, whatever the state of class 1. In that case, $\bar{\phi}_1$ takes a slightly simpler form. After simple calculations:

$$\bar{\phi}_1(cz) = H(z) = \frac{z\int_0^{\rho_2} u^{z-1}\exp(u)\,\mathrm{d}u}{\rho_2^z \exp(\rho_2)}.$$

Using the monotonicity of ϕ_1 in its first variable, we can conclude that $u_1(t)$ converges monotonically to its limit point, and that

$$\bar{u}_1 = \begin{cases} u_1(0), & \text{if } \lambda_1 < \bar{\phi}_1(u_1(0)); \\ \bar{\phi}_1^{-1}(\lambda_1), & \text{otherwise,} \end{cases}$$

The inequality (2.7) can be ensured by scaling the process $u_1(t)$ by a factor $\bar{z}/\bar{u}_1$, which, in turn, can be achieved by scaling the priority level (or, equivalently, r_1) by this very same factor. This additional scaling results in a larger share of the bandwidth for class-1 flows in case $\bar{z} > \bar{u}_1$. Conversely, if $\bar{z} < \bar{u}_1$, the priority level of class-1 flows is appropriately decreased so that the blocking probability constraint of class-2 flows is not violated.

CHAPTER 3

QUEUE WITH POISSON CONTROLLER

In the previous two chapters, we looked at models in which either all or a subset of classes were scaled. In the latter case, the subset which was scaled did change over time. In this chapter, we shall study a two-dimensional Markov chain in which one of the two dimensions needs to be scaled. However, this time the dimension that is scaled changes over time.

This type of models arise in the context of real-time control or optimization of communication network. In several of these problems, the policy depends upon the state of the network (see [9] for a variety of control problems of this kind). For example, the bandwidth-allocation policies in the previous chapter were based on the number of ongoing flows of each class in the network. In practice, this information may be costly to obtain in real-time, especially in big networks with a large number of concurrent flows and where the state can change at very short time-scales. In such situations, it may nevertheless be possible for the decision-maker to observe the state of the network every once in a while and change the policy appropriately. The dynamics of the network is now modelled by the current state and the state observed at the previous control instant. If the control instants are not planned sufficiently often enough, then the current state and the last observed state may become completely out of phase. In the terminology of the previous chapter, when the current state experiences a surge, the last observed state can remain on the normal scale and vice versa.

In this chapter, we shall study one such model which exhibits time-varying change in scales. It a simple model of a single-server queue in which speed of the server is adjusted at Poisson instants that are independent of the state of the system. We call this model with Poisson control instants as *Poisson controller*. It has been shown that [55, 56], when the objective is to minimize a linear combination of mean energy consumption and mean sojourn time, the optimal server-speed is proportional to

the number of customers to the power of a constant. This achieves the right-tradeoff between the mean sojourn time and mean energy consumption which are opposites in the sense that reducing the sojourn time requires increasing the speed which increase the energy consumption and vice versa. Inspired from this policy, it will be assumed that the speed is set to $\min(q, \bar{s})$ when q is the observed queue length and $\bar{s}$ the maximum server speed. Although the actual optimal policy requires the exponent to larger than 1, we take it to be 1 for ease of analysis although similar behaviour will be also observed for other values of the exponent.

From the practical point of view, our results can be used to compute the difference in the performance metrics due to control instants being different from the ones prescribed by the optimal policy. Of course, one could actually compute the optimal policy for the model with a Poisson controller and implement it. However, since there is quite a vast literature and results assuming state is known instantly, we first consider the case when a policy computed for the setting of instantaneous information in implemented with a Poisson controller,

Contributions and related work

In [NQPR23], we analyse the Poisson controller model described by a two-dimensional Markov process with state given by $(Q(t), S(t))$ where $Q(t)$ denotes the number of customers in the system at time t and $S(t)$ denotes the speed of the server at t. The two cases $\bar{s} < \infty$ and $\bar{s} = \infty$ are treated separately since their analysis is based on different techniques. Moreover, as will be seen later, they give rise to different asymptotic results.

For the infinite maximum speed case, that is $\bar{s} = \infty$, we determine the functional equation whose solution leads to the joint generating function of the steady-state process. The steady-state distribution of the queue-length conditional on the speed is shown to be equivalent to the transient distribution of a queue whose state is reset to the given speed at the control instants. The latter queue was analysed in [57] in which the generating function for the steady-state queue-length was given. From this conditional generating function as well as the observation that the marginal distribution of Q is the same as that of S, we obtain a system of linear equations to compute this marginal distribution. Finally, we investigate the limiting behaviour of

the steady-state distribution when the rate of the control process, ν, goes to 0 and to ∞. For $\nu \to 0$, it is possible that the queue is unstable for some of the speeds. The queue-length can thus live on two different scales: (i) the fluid scale when the sampled speed is less than the arrival rate; (ii) the normal scale when the sampled speed is greater than the arrival rate.

For finite maximum speed $\bar{s}$, we resort to the matrix geometric method [58] to analyze the system. Using a probabilistic interpretation, we obtain an explicit expression for the R matrix. We show how to obtain the joint generating function when $\nu \to 0$ and $\nu \to \infty$.

Our model is related to queueing systems or Markov chains in random environments [59, 60]. In the cited models, the arrival rate or the service rate of the queue depends upon the state of the environment which is a random process. As opposed to this, in our model, the environment (which is the speed observed by the controller) itself depends upon the queue length since it is set to the value measured at the control instants. The two variables – the queue-length and the state – thus influence the dynamics of each other in our model whereas in the classical random environment model it is only the environment that influences the dynamics of queue-length. We also mention [61] in which a random environment model with both unstable and stable speeds was analysed. They obtained the conditional generating function for the queue-length for the process on both the fluid as well as the normal scale. We obtain this type of results for the Poisson controller model.

3.1 Poisson controller with infinite maximum speed

Consider a single-server queue to which arrivals occur according to a Poisson process of rate λ. Each arrival brings with it an exponentially distributed service requirement with mean $1/\mu$. The speed of the server can be dynamically assigned values in the set $\{0,1,2,\ldots,\}$. Adjustments to the speed can be made only at control instants which are assumed to occur according to a Poisson process of rate ν independently of the arrivals and the departures. At a control instant, the speed of the server is set equal to the number of customers observed at that instant. Between any two consecutive control instants, the speed of the server remains constant at the value

chosen at the earlier control instant.

Let $Q(t)$ denote the number of customers in the system at time t and $S(t) = Q(\tau_t)$, with τ_t the last control instant at or before time t. Thus, the speed at time t is maintained at $Q(\tau_t)$ until the next control instant. As our focus is on the number of customers in the system and service times are exponential, the service discipline can be any discipline in which the service order is independent of the actual service times of the customers (e.g., FCFS, LCFS or ROS).

The process $(Q(t), S(t))_{t\geq 0}$ is a Markov process with transition rates

$$(Q(t), S(t)) \rightarrow \begin{cases} (Q(t)+1, S(t)) & \text{with rate} \quad \lambda; \\ (Q(t)-1, S(t)) & \text{with rate} \quad \mu S(t); \\ (Q(t), Q(t)) & \text{with rate} \quad \nu. \end{cases} \tag{3.1}$$

The process $(Q(t), S(t))_{t\geq 0}$ is ergodic for all possible combinations $\lambda > 0$, $\mu > 0$ and $\nu > 0$ and we are interested in the steady-state behaviour of this two-dimensional Markov process. Denote with $\pi_{i,j} = \lim_{t\to\infty} P(Q(t) = i, S(t) = j)$ the steady-state probabilities and let

$$P(x, y) = \sum_{i\geq 0, j\geq 0} \pi_{i,j} x^i y^j \tag{3.2}$$

be the corresponding joint probability generating function. By (Q, S) we denote a pair of random variables with this joint probability generating function.

Theorem 8. *$P(x, y)$ is the solution of the functional equation*

$$(\nu + \lambda(1-x))P(x, y) + \mu y\left(1 - \frac{1}{x}\right)\frac{\partial}{\partial y}\left[P(x, y) - P(0, y)\right] = \nu P(xy, 1). \tag{3.3}$$

Substituting $x = 1$ in the above theorem, we get:

Corollary 4. *The marginal distribution of the speed is the same as the marginal distribution of the number of customers in the system. That is,*

$$P(1, y) = P(y, 1).$$

An explanation for Cor. 4 is that the controller sees the marginal distribution of Q due to the PASTA property. Since S is set to Q at the control instants, the marginal distribution of S just after these instants is the same as the marginal distribution of Q. Because the time until the next control instant is independent of S at a control instant (this time is exponentially distributed with parameter ν), the marginal distribution of S at an arbitrary instant is the same as the marginal distribution of S just after a control instant and hence also the same as the marginal distribution of Q at an arbitrary instant.

We can also conclude that the expected queue-length is larger than in an $M/M/\infty$ queue, i.e., in a queue when the control is based on instantaneous information.

Corollary 5.

$$\mathbb{E}[Q] > \rho, \text{ with } \rho = \lambda/\mu.$$

An explicit solution to the functional equation (3.3) was not within our reach. So, we provide an alternative way to compute some of the performance measures such as the mean queue-length. This method is not completely explicit but maybe more amenable to numerical computations. For this we shall obtain expressions for both the queue-length distribution conditioned on the server-speed being $j\mu$ as well as for the marginal distribution of the server-speed. Combining these quantities, the marginal performance measures of the queue-length can be computed. The small footprint in this method is that the marginal distribution of the server-speed is in terms of an infinite system of linear equations.

The conditional queue-length distribution

Define $\sigma_j = \sum_i \pi_{i,j}$ to be the marginal distribution of the stationary server speed. Let

$$p_j(i) = \frac{\pi_{i,j}}{\sigma_j}$$

be the stationary conditional probability of having i customers in the system when the service rate is $j\mu$. Further, define

$$\beta_j(\nu) = \frac{\lambda + j\mu + \nu - \sqrt{(\lambda + j\mu + \nu)^2 - 4\lambda(j\mu)}}{2\lambda}, \tag{3.4}$$

and $\tilde{\beta}_j(\nu) = \lambda\beta_j(\nu)(j\mu)^{-1}$.

Theorem 9. *The conditional queue-length distribution is given by*

$$p_0(l) = \frac{\nu}{\nu+\lambda}\left(\frac{\lambda}{\nu+\lambda}\right)^l, \tag{3.5}$$

and, for $j > 0$,

$$p_j(l) = \frac{\nu}{\lambda}\sum_{k=0}^{l} c_{k,j}\tilde{\beta}_j(\nu)^{l-k+1}. \tag{3.6}$$

where

$$c_{k,j} = \begin{cases} (1-\beta_j(\nu))^{-1}\beta_j(\nu)^j & k = 0; \\ \beta_j(\nu)^{j-k} & k = 1,\ldots,j; \\ 0 & k > j. \end{cases} \tag{3.7}$$

Remark 2. *The stationary distribution of Q_j is the same as the transient distribution after an exponentially distributed time period with parameter ν in an $M/M/1$ queue with arrival rate λ, service rate $j\mu$ and starting at time 0 in state j. Hence, the above result can be alternatively derived using results on the transient distribution in the $M/M/1$ queue (see Section I.4.4 and in particular formula (4.27) in [57]).*

The unconditional measures can then be obtained from the marginal distribution of the server-speed given in the following result.

Proposition 3.

$$\sigma_l = \sum_j \sigma_j \frac{\nu\tilde{\beta}_j(\nu)}{\lambda}\left(\sum_{k=0}^{l} c_{k,j}\tilde{\beta}_j(\nu)^{l-k}\right), \forall\, l \geq 0. \tag{3.8}$$

Equation (3.8) can be interpreted as the balance equation of the embedded Markov chain of the server speed at control instants.

Asymptotics for $\nu \to \infty$.

When the rate of control $\nu \to \infty$, intuitively one expects the speed to be the same as the queue length since measurements are being made at a much faster rate compared to rates of variations in the queue-length. The two-dimensional process will live mainly on the diagonal states. The following result formalizes this intuition.

Proposition 4. *If $\nu \to \infty$, then $P(x,y) \to e^{\rho(xy-1)}$.*

Asymptotics for $\nu \to 0$

On the other extreme, when the measurements are performed at a much slower rate, time-scale separation between the queue-length and the server-speed occurs. Since the server-speed does not change between two control instants, the queue-length evolves on a faster time-scale compared to the server-speed. In the spatial dimension, both the queue-length and the server-speed can evolve on two scales: fluid scale on which they are $O(\nu^{-1})$ and the normal scale on which they are $O(1)$. The trajectories on these processes rescaled by ν will be cyclic as shown in Fig. 3.1. We see the out-of-phase trajectories of the queue-length and the server-speed. When one is on the fluid scale the other one is on the normal scale and vice versa.

The intuitive reasoning for this behavior is as follows. Assume time has been rescaled by a factor ν so that control instants form a Poisson process of rate 1. A cycle begins when the measured queue-length is 0. For this speed, the queue-length process being unstable, it grows linearly on the fluid scale at rate λ until the next measurement instant. That is, during this period, the limiting process $\lim_{\nu\to 0} \nu Q(t)$ will grow linearly. The queue-length being on the fluid scale, the next measurement will set the speed to a value $O(\nu^{-1})$. This will bring the queue-length to 0 instantaneously in time $O(\nu)$. The speed is now $O(\nu^{-1})$ whereas the arrival rate is λ. So, the queue-length will remain at 0 (on the normal scale and not just on the fluid scale) until the next measurement instant at which point the speed will be set to 0. At this point, a new cycle will begin. We were unable to formalize this intuition and leave it a conjecture.

Define $\hat{P}(x,y) := \lim_{\nu\to 0} P(x^\nu, y^\nu)$ to be the generating function of the scaled pair of random variables $(\nu Q, \nu S)$ in the limit $\nu \to 0$.

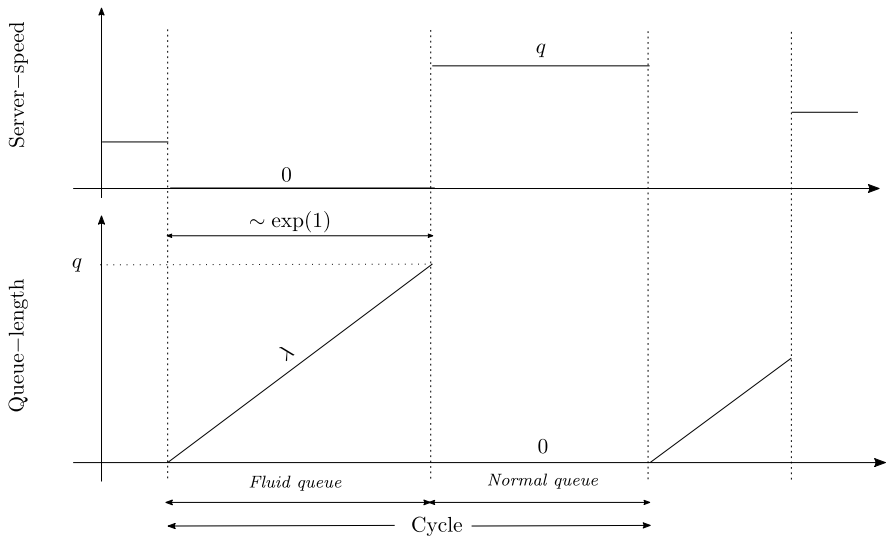


Figure 3.1 – Trajectories of the rescaled queue-length and the rescaled server-speed in the limit $\nu \to 0$. The vertical dashed lines are control instants.

Conjecture 1.

$$\hat{P}(x,y) = \frac{1}{2}\frac{1}{1-\lambda\log(x)} + \frac{1}{2}\frac{1}{1-\lambda\log(y)}. \tag{3.9}$$

As $\nu \to 0$, the probability mass of the joint process concentrates around the two axes $Q = 0$ and $S = 0$. When Q is on the fluid scale, $S = 0$ while when $Q = 0$, S is on the fluid scale. On each of these two axes the scaled stationary process behaves like an exponentially distributed random variable of rate λ^{-1}. The coefficient $1/2$ for each of the two terms in the above generating function corresponds to the proportion of time spent by the process on each of the two axes.

3.2 Poisson controller with finite maximum speed

Two observations can be made from the trajectories in Figure 3.1: one is the out-of-phase dynamics of the two dimensions, and the second one is that the duration of each phase is itself random since it lasts for an exponentially distributed time of rate 1 (on the macroscopic scale). In this section, we look at the model with finite

maximum speed $\bar{s}$ in which the second observation becomes more pronounced. We shall see that in additional to the cycle duration, even the slope of the fluid trajectories becomes random. Of course, since the maximum speed is finite, the speed itself remains always on the normal scale. So, we will not see the out-of-phase dynamics of the previous section.

The process $(Q(t), S(t))_{t\geq 0}$ is now a Quasi-Birth-Death (QBD) process (see [62], *e.g.*) with level-dependent transition rates. Furthermore, for $Q(t) \geq \bar{s}$ the transition rates become level-independent. It can be argued that the ergodicity condition of this process is given by the natural condition $\lambda < \bar{s}\mu$, for every $\nu > 0$.

From the theory of QBD process, the stationary probability vector $\pi_n = [\pi_{n,0}, \pi_{n,1}, \ldots, \pi_{n,\bar{s}}]$ for the different states with n customers in the system, are, for $n \geq \bar{s}-1$, of the form

$$\pi_n = \pi_{\bar{s}-1} R^{n-\bar{s}+1}, \tag{3.10}$$

where R is the minimal non-negative solution of a matrix equation which depends on the transition rates.

While in general the R matrix does not have a nice formula, for our model, this matrix can be explicitly calculated. It is a diagonal matrix with additional non-zero values in the last column.

Proposition 5. *The matrix R is given by*

$$R = \begin{pmatrix} \frac{\lambda}{\lambda+\nu} & & & & & \frac{\lambda}{\bar{s}\mu} \\ & \frac{\lambda\beta_1(\nu)}{\mu} & & & & \frac{\lambda(1-\beta_1(\nu))}{\bar{s}\mu} \\ & & \ddots & & & \vdots \\ & & & \frac{\lambda\beta_i(\nu)}{i\mu} & & \frac{\lambda(1-\beta_i(\nu))}{\bar{s}\mu} \\ & & & & \ddots & \vdots \\ & & & & & \frac{\lambda}{\bar{s}\mu} \end{pmatrix}. \tag{3.11}$$

Since R is of the above form, R^n too has a simple formula which can be used for computing π_n in (3.10). The probability vectors π_n, $n \leq \bar{s}-1$, can now be computed using a system of linear equations and the normalization equation (see, e.g., [58]).

Further, the joint generating function can be expressed as

Proposition 6.

$$P(x,y) = \sum_{n<\bar{s}-1,j} \pi_{n,j} x^n y^j + \pi_{\bar{s}-1}(I-Rx)^{-1} x^{\bar{s}-1} \cdot \mathbf{y}^T, \tag{3.12}$$

with $\mathbf{y} = [1, y, y^2, \ldots, y^{\bar{s}}]$

We now look at the asymptotics when $\nu \to \infty$ and $\nu \to 0$. The former regime is less interesting as was the case in Sec. 3.1. So, we only present the case $\nu \to 0$.

Asymptotics for $\nu \to 0$

In the following, we shall assume that time has been rescaled by a factor ν so that the control instants happen according to a Poisson process of rate 1. Define

$$\mathscr{S}^+ = \{j : j > \rho\}, \quad \mathscr{S}^- = \{j : j \leq \rho\}. \tag{3.13}$$

The set $\mathscr{S}^+$ contains the speeds for which the queue is positive recurrent while $\mathscr{S}^-$ is its complement set.

As $\nu \to 0$, the trajectories of the queue-length will converge to one described in the following (the intuitive argument is based on time-scale separation but we do not have a formal proof). Let a cycle denote the time between two consecutive observations in $\mathscr{S}^-$. An illustration of trajectories of the *queue-length on the fluid scale* and the speed is shown in Fig. 3.2. Until the first control instant after the start of a cycle, the queue-length grows linearly on the fluid scale with rate $\lambda - j\mu$ where j is the queue-length sampled at the start of the cycle. Hence, at the first control instant the speed will necessarily be set to $\bar{s}$, which is in $\mathscr{S}^+$. Since the queue-length process is now stable, it will decrease with rate $\bar{s}\mu - \lambda$ for one or more control instants until the fluid hits 0. (Notice that control instants do not affect the server working at constant speed $\bar{s}$ while the queue is at the fluid level, ensuring that level 0 will be reached.) The queue-length process will now evolve on the normal scale for one or more control instants until a speed from $\mathscr{S}^-$ is sampled. At this point, a new cycle will begin. A cycle can thus be decomposed into three phases: the *fluid unstable* phase during which the fluid grows; the *fluid stable* phase during which the fluid drains; and the n$^{\text{o}}$ phase during which the queue length lives on the normal scale.

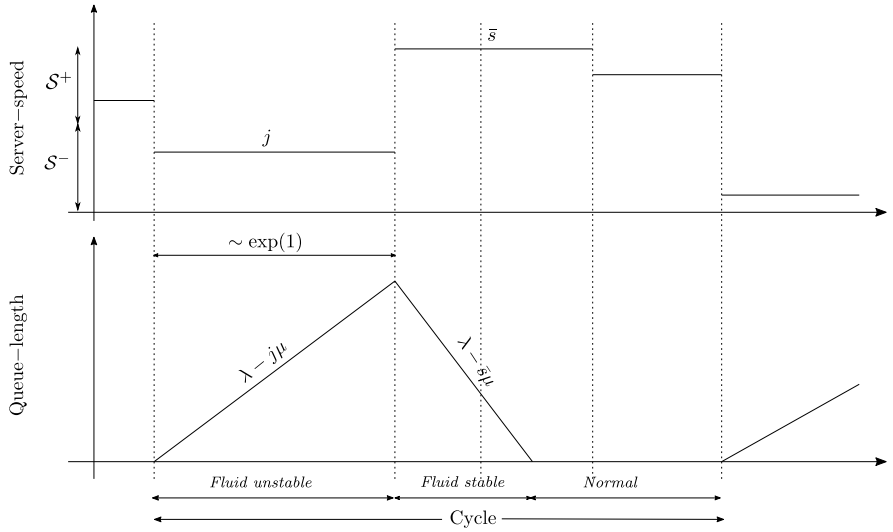


Figure 3.2 – Trajectories of the queue-length on the fluid scale and the server speed in the limit $v \to 0$. The vertical dashed lines are control instants. Trajectories that occur on the normal scale all collapse to 0.

As in previous section, in the spatial dimension, the queue-length process can be either on the fluid scale or on the normal scale depending on whether the sampled server-speed was smaller or larger than λ. However, unlike in the infinite maximum speed case, here the speed always remains on the normal scale. Another difference with the previous section is that the slopes of the trajectories in random chosen in each cycle.

The above arguments allow us to obtain the to compute the generating function $P(x,y)$ on the different scales. Define $\hat{P}(x,y) = \lim_{v\to 0} P(x^v, y)$ for the generating function that captures the queue-length on the fluid scale and $\tilde{P}(x,y) = \lim_{v\to 0} P(x,y)$ to be the one that captures the normal scale queue-length.

$$\hat{P}(x,y) = \sum_{j\in\mathscr{S}^-} \frac{\sigma_j}{1-(\lambda - j\mu)\log(x)} \left(y^j + \frac{\lambda - j\mu}{\bar{s}\mu - \lambda} y^{\bar{s}} \right) + \sum_{j\in\mathscr{S}^+} y^j \sigma_j, \tag{3.14}$$

$$\tilde{P}(x,y) = \sum_{j\in\mathscr{S}^+} \frac{1-\rho_j}{1-\rho_j x} y^j \sigma_j. \tag{3.15}$$

The marginal distribution σ_j can be computed from the stationary distribution of

the Markov chain embedded at the control instants and the renewal reward theorem. Again in this asymptotic the expressions for the generating function simplifies (assuming we are able to formalize the time-scale separation argument.)

Remark 3. *The above asymptotic analysis cannot be used to obtain the results in Sec. 3.1 for the infinite speed case by taking $\bar{s} \to \infty$. The difficulty comes from the fact that when $\bar{s} = \infty$, the speed sampled after the fluid (unstable) phase is an exponential random variable. On the other hand, in the finite $\bar{s}$ case, this speed is always a fixed value. The limit of this sequence of deterministic values is unable to capture the exponential random variable at $\bar{s} = \infty$. One will have to scale $\bar{s}$ as ν^{-1} to get back the results of Sec. 3.1.*

3.3 Example application

An application of our analysis can be to quantify the trade-off between the monitoring costs and the suboptimality due to reduced monitoring. As mentioned earlier, control policies are sometimes obtained assuming changes can be made at arbitrary time instants (or equivalently assuming infinite monitoring frequency). Monitoring the state at high frequencies can however be costly in terms of resources (monitoring packets consume bandwidth, etc.). Reducing the monitoring frequency lowers the measurement cost but also makes the policy more suboptimal. With the analysis in this chapter, one can compute the performance obtained for a given monitoring frequency, ν, and then determine the appropriate value of ν that optimizes the objective that accounts for both the performance as well as the monitoring costs.

As an illustration, consider the problem of optimizing a linear combination of sojourn time and energy consumption that is mentioned in the introduction of the paper. In Fig. 3.3, the expected queue length as well as the expected speed as a function of ν, the monitoring frequency, is plotted for $\bar{s} = 2$. It can be seen that both these quantities decrease with ν which points to a trade-off since a higher ν will entail a higher monitoring cost but lower energy and delay costs.

We remark that for the finite-speed case Prop. 6 can be generalized to arbitrary increasing function speed profiles s_i, where s_i is the prescribed speed when state i is observed. Similar analysis can thus be performed on other specific speed profiles.

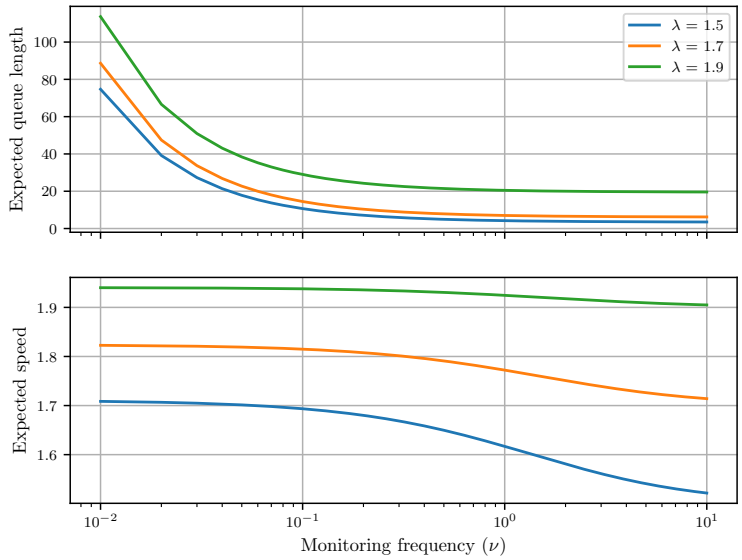


Figure 3.3 – Expected queue length and expected speed as functions of the monitoring frequency (ν). $\bar{s} = 2$, $\mu = 1$.

PART II

Distributed systems

CHAPTER 4

SINGLE-PATH ROUTING

In this part, we turn our attention to distributed systems with multiple entities simultaneously optimizing their own objectives while sharing a common network and influencing the performance of each other. This chapter concerns itself with solving a centralized problem using a distributed architecture. Such an approach could be followed for reasons of scalability and robustness.

The single-path routing problem, also known as the unsplittable or non-bifurcated multicommodity flow problem, naturally arises in a variety of contexts. It amounts to routing a given set of fixed traffic demands in a network, allocating each demand to a single path to minimize the total network cost, which is usually expressed as the sum of link costs (see Chapter 8 of [63] for the related joint routing and congestion control problem). Our motivation for studing this problem comes from the Multi-Protocol Label Switching (MPLS) technology [64] that changed the usual hop-by-hop paradigm by enabling network flows to be routed along predetermined paths. In an MPLS network, each flow is routed along a single path, but two different flows with the same source/destination pair can use two different paths. Traffic engineering in MPLS networks mainly amounts to optimizing the quality of service of network flows by tailoring the paths assigned to flows to the prevailing traffic conditions.

The network model is similar to the bandwidth-sharing networks of Chapter 2. Here, the bandwidth-sharing policy is assumed to be given and we are interested in computing the best route for the flows. The link cost function (latency, *e.g.*) is assumed to be obtained from performance analysis done previously on the bandwidth-sharing policy in place.

Contributions and related work

The single-path routing problem belongs to the class of non-linear mathematical programs involving integer variables (actually, binary variables). These mathemati-

cal programs are known to be extremely hard to solve, both from a theoretical and from a practical point of view (see Chapter 15 of [65] for a survey as well as [66] and references therein for a thorough literature review). Even in the simplest case with binary variables, quadratic function and equality constraints, they are known to be NP-hard [67]. Several general approaches have been proposed to solve non-linear integer programming problems. Some transform the discrete problem into a continuous one (see for example [68]). Despite their qualities, those techniques do not scale very well with the size of the problems. A recent alternative is the so-called *Global Smoothing Algorithm* [69], which seems to scale better while providing fairly good approximations. Heuristics and meta-heuristics have also been used to find an approximate solution to non-linear integer programming problems. Among others, ant-inspired optimization techniques are known to be efficient for solving various routing problems [70, 71].

In [BPV17], we propose an approximation algorithm, inspired from game theory [72], for solving non-linear single-path routing problems with additive link costs. Instead of the network choosing a path for the flows, we let the individual flows to independently select their path to minimize their own cost function. However, the cost functions of the flow are designed so that the resulting Nash equilibrium of the game provides an efficient approximation of the optimal solution. We note that a similar algorithm was proposed in [73] for scheduling of strictly periodic tasks. We establish the convergence of the algorithm and show that every optimal solution is a Nash equilibrium of the game. We also prove that if the link latency functions ℓ_e are polynomial of degree $d \geq 0$, then the approximation ratio of the algorithm is $\left(2^{1/(d+1)}-1\right)^{-(d+1)}$. As will be shown numerically, the main merit of this algorithm is that it is several orders of magnitude quicker than the method discussed above while providing good optimization results.

Problem Statement

Consider a network represented by a directed graph $G = (V, E)$. To each edge $e \in E$ is associated a non-decreasing latency function $\ell_e : \mathbb{R}_+ \to \mathbb{R}_+$ which can interpreted as the delay per packet. A typical example is the M/M/1 delay function $\ell_e(x) = 1/(c_e - x)$, where c_e represents the capacity of link e.

For any set $\pi \subset E$, we define the constant δ_π^e as 1 if $e \in \pi$, and 0 otherwise. We are given a set $\mathcal{K} = \{1, 2, \ldots, K\}$ of Origin-Destination (OD) pairs with $\lambda_k \in \mathbb{R}$ being its traffic demand. Each traffic demand has to be routed in the network over a single path. Let Π_k be the set of all paths for the OD pair k. This set can contain all simple paths from s_k to t_k, or only a subset of those paths satisfying some constraints. A single-path routing strategy is a vector $\boldsymbol{\pi} = (\pi_k)_{k \in \mathcal{K}} \in \Pi$, where π_k is the path assigned to traffic demand k and $\Pi = \Pi_1 \times \Pi_2 \times \cdots \times \Pi_K$. The goal is to find a routing strategy that minimizes the cost of the network $F(\boldsymbol{\pi}) = \sum_{e \in E} y_e(\boldsymbol{\pi}) \ell_e(y_e(\boldsymbol{\pi}))$, where $y_e(\boldsymbol{\pi}) = \sum_{k \in \mathcal{K}} \delta_{\pi_k}^e \lambda_k$ is the total traffic flowing on link e in routing strategy $\boldsymbol{\pi}$.

Formally, the problem can be written as a 0-1 mathematical program as follows:

$$\text{minimize} \sum_{e \in E} y_e \ell_e(y_e) \tag{4.1}$$

subject to

$$y_e = \sum_{k \in \mathcal{K}} \sum_{\pi \in \Pi_k} \lambda_k \delta_\pi^e x_{k,\pi} \qquad e \in E, \tag{4.2}$$

$$\sum_{\pi \in \Pi_k} x_{k,\pi} = 1 \qquad k \in \mathcal{K}, \tag{4.3}$$

$$x_{k,\pi} \in \{0, 1\} \qquad \pi \in \Pi_k, k \in \mathcal{K}. \tag{4.4}$$

4.1 Penalized best-response algorithm

The best-response algorithm is used in game theory as the reaction of a rational player to the actions of the other players. The best-response of a player is defined as action that minimizes its objective function conditioned on the actions of the other players. It is, as the name suggests, the best that the player can do for a given action profile of the others. The best-response algorithm then consists of players taking turns in some order to adapt their actions based on the most recent known actions of the others until an equilibrium point, known as the Nash equilibrium (NE), is reached.

A naive algorithm would be to assign each OD pair the role of player with the

objective of minimizing its own delay

$$f_i(\pi, \boldsymbol{\pi}_{-i}) = \sum_{e \in \pi} \lambda_i \ell_e(y_e(\pi, \boldsymbol{\pi}_{-i})), \quad \forall \pi \in \Pi_i. \tag{4.5}$$

This approach has two drawbacks due to the fact that Prob. 4.1 is not a game with different and conflicting objectives for each flow. Rather the flows have a common objective function. The first drawback of the naive approach is that a Nash equilibrium of this game need not be the optimum of the (4.1) (see Example 1).

Example 1. *Consider a network with two parallel links as shown in Figure 4.1. There are two flows, each of size* 1*, which wish to go from O to D. There are two possible routes: (i) the top-route with a latency function of* $l(x) = x$*; and (ii) the bottom-route with a latency function of* $l(x) = 0.4x$*.*

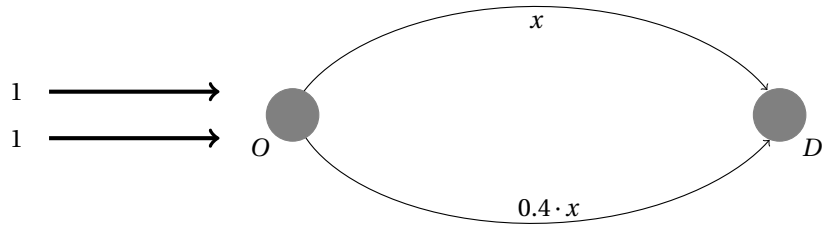


Figure 4.1 – Example to show that the Nash equilibrium of a weighted congestion game need not be a global optimum.

It can be verified that at the global optimum one of the flows will be routed through the top-route and the other will be routed through the bottom-route. The cost for the flow on the top-route at the optimal routing is 1 *while for the one on the bottom-route it is* 0.4*. In order to check that the optimum is not a Nash equilibrium for cost defined in* (4.5)*, we look at the best-response of the flow on the top-route. If this flow moves to the bottom-route its cost will be* $0.4(1+1) = 0.8$ *which is less than its current cost of* 1*. Thus, at the global optimum the flow on the top-route has an incentive to move to the bottom-route.*

And, secondly, the convergence of best-response algorithm is known only in specific cases which does not include the cost function (4.5) derived from the single-path routing problem. Since our aim is to design a fast approximation algorithm for solving (4.1), convergence guarantee is desirable. Further, if the set of Nash equilibria

of an algorithm contains the global optimum, it would mean that once in while the algorithm will find the optimum solution which could be desirable as well.

An intuitive reason NE is not necessary a global optimum for the total cost function is that players minimize their individual objective without accounting for the impact of their actions on the others. While the best-responding player reduces its cost, the others can potentially see an increase in their costs. Due to the non-zero sum nature of the game, the total cost can be larger after a best-response even though the cost of the best-responding player is reduced. Example 1 also illustrates this point. At the global optimum, when the flow on the top-route moves to the bottom one, its cost reduces by 0.2 units while that of other player increases by 0.4.

To overcome these two difficulties, we follow the approach proposed in [74]. The key idea is to modify the cost function of each player in such a way that the two properties are satisfied. This can be done by adding a penalty corresponding to the impact of a player's action on the cost of the other players. When player i routes its traffic on path π, the increase in the cost of each player $j \neq i$ using link $e \in \pi$ is

$$\lambda_j \left[\ell_e \left(y_e^{-i} + \lambda_i \right) - \ell_e \left(y_e^{-i} \right) \right], \tag{4.6}$$

where $y_e^{-i} = \sum_{k \neq i} \delta_{\pi_k}^e \lambda_k$ represents the total traffic flowing on link e due to all players other than i. As a consequence, we define the penalty term as follows:

$$\begin{aligned} p_i(\pi, \boldsymbol{\pi}_{-i}) &= \sum_{e \in \pi} \sum_{j \neq i} \lambda_j \delta_{\pi_j}^e \left[\ell_e \left(y_e^{-i} + \lambda_i \right) - \ell_e \left(y_e^{-i} \right) \right], \\ &= \sum_{e \in \pi} y_e^{-i} \left[\ell_e \left(y_e^{-i} + \lambda_i \right) - \ell_e \left(y_e^{-i} \right) \right]. \end{aligned} \tag{4.7}$$

In the penalized best-response (PBR) algorithm, player i computes its path to minimize its penalized cost:

$$\text{minimize}_{\pi \in \Pi_i} c_i(\pi, \boldsymbol{\pi}_{-i}) = f_i(\pi, \boldsymbol{\pi}_{-i}) + p_i(\pi, \boldsymbol{\pi}_{-i}). \tag{OPT-i}$$

As shown in [74], the convergence of PBR in finite-time follows from the fact the objective in (4.1), F, is a potential function for this game, *i.e.*, a decrease in the individual utility leads to a corresponding decreasing in the global objective function, F.

Further from the same argument it can also be shown the global minimum of (4.1) is a Nash equilibrium of the penalized game.

Remark 4. *The worst-case computational complexity of finding a pure Nash equilibrium in a potenial game can be exponential in the number of players (the number of flows in our problem)[75]. However, in [76] it is shown that on average the complexity is linear in the number of players. Thus, on random instances, the best-response algorithm can be expected to converge much faster than exact algorithms.*

Approximation ratio of PBR

An instance I of our problem is defined by the graph $G = (V, E)$, by the set of traffic demands $\mathcal{K}$ (including the demand value λ_k and the set of candidate paths Π_k for routing each demand k) and by the link latency functions $\ell_e : \mathbb{R}_+ \to \mathbb{R}_+$. Our goal in this section is to establish bounds on the approximation ratio of PBR that hold uniformly over all instances of the problem. More precisely, given an instance I of the problem, let $\boldsymbol{\pi}$ be any Nash equilibrium and let $\boldsymbol{\pi}^*$ be an optimal routing strategy for that instance. We look for an upper bound on the ratio $F(\boldsymbol{\pi})/F(\boldsymbol{\pi}^*)$ that holds for all instances I of the problem. In the following, we establish such a bound when the link latency functions are polynomial functions of the form $\ell_e(x) = \sum_{j=0}^{d} a_{e,j} x^j$, where $d \geq 0$ and the a_e are non-negative coefficients. Our main result is stated in Theorem 10.

Theorem 10. *If $\ell_e(x) = \sum_{j=0}^{d} a_{e,j} x^j$, then the approximation ratio of the penalized best-response algorithm is $\left(2^{1/(d+1)} - 1\right)^{-(d+1)}$.*

Interestingly, in the case $d = 0$, that is, when the latency function $\ell_e(x)$ of each link e is a constant $a_e \geq 0$, the penalized best-response algorithm is guaranteed to provide an optimal solution.

The proof has two parts – one which shows that the ratio is an upper bound, and the other which proves that this is a lower bound. For the upper bound, Hölder inequality is used to bound the difference between the Nash equilibrium and deviations from it, as was done by [77] to compute an upper bound for the PoA of the standard best-response (SBR) algorithm (the one when the objective is (4.5)) for affine and polynomial cost functions with non-negative coefficients.

In order to show prove the lower bound, we provide an instance which attains this upper bound. It is adaptation of the instance given in [78] (see Lower Bound 2 and Lemma 4.5 in that paper). Consider a problem with cost function $\ell_e(x) = a_e x^d$, and N origin-destination pairs O_i-D_i. In addition to these nodes, there are two other nodes R_0 and R_1 which act as routers. Each origin has two choices: either send the traffic to the left or to the right.

Let the edge connecting O_i and O_{i+1} be numbered i, and set the coefficient of this link, $a_i = \phi^{(d+1)i}$. The coefficient of the link $R_0 - O_1$ is set to 1 and that of $O_4 - R_1$ is set to $a_N = \phi^{(d+1)(N-1)}$, where $\phi = 2^{1/(d+1)} - 1$. All the other edges have a cost of 0. Figure 4.2 shows an instance with $N = 4$ pairs.

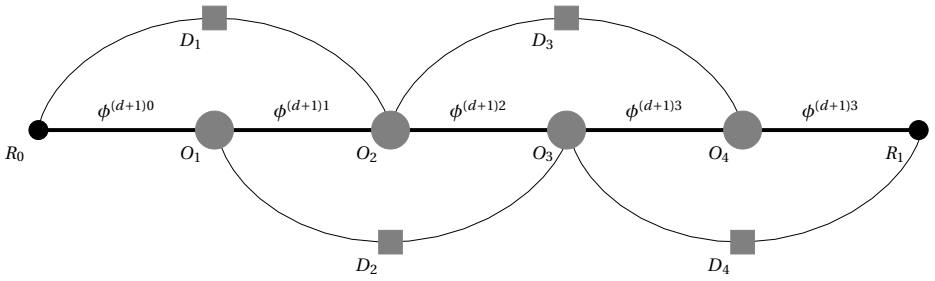


Figure 4.2 – Problem instance with $N = 4$ origin-destination pairs for the lower bound on the approximation ratio.

For this instance, the routing to the left corresponds to a Nash equilibrium whereas routing to the right corresponds to the global optimum. As a consequence, the approximation ratio is

$$g_N(\phi) = \frac{N}{(N-1)\phi^{d+1} + 1}. \tag{4.8}$$

By taking $N \to \infty$, we obtain an approximation ratio of $\phi^{-(d+1)} = (2^{1/(d+1)} - 1)^{-(d+1)}$ as claimed.

An immediate consequence of Theorem 10 is that:

Corollary 6. *As $d \to \infty$, the approximation ratio is asymptotically $\left(\frac{d+1}{\log(2)}\right)^{d+1}$.*

In Table 4.1, we compare the approximation ratio of PBR to that of SBR. For the latter, the game is a weighted congestion game (see [78] and references therein) for

which several authors, including [78] and [79], have investigated the PoA. For polynomial cost functions of degree d, it was shown in [79] that the PoA for mixed equilibrium is Φ_d^{d+1}, where Φ_d is the solution of $(\Phi_d+1)^d = \Phi_d^{d+1}$. When $d \geq 1$, the approximation ratio of the PBR is worse than that of SBR with asymptotically, the standard version being a factor $(\log(d))^{d+1}$ better than the penalized version. On the downside, SBR does not have guaranteed convergence whereas PBR is guaranteed to converge.

It is however important to keep in mind that the above performance guarantees are obtained for worst-case scenarios that are not necessarily representative of instances met in practice as shall see next.

Table 4.1 – Comparison of the approximation ratios of PBR and SBR for different values of d.

degree (d)	Penalized BR	Standard BR [79, 78]
0	1	1
1	5.83	2.62
2	56.95	9.91
3	780.28	47.82
asymptotic	$\Theta\left(\left(\frac{d+1}{\log(2)}\right)^{d+1}\right)$	$\Theta\left(\left(\frac{d}{\log(d)}\right)^{d+1}\right)$

4.2 Numerical results

We give a sample of results with the link costs taken to be the $M/M/1$ delay function. PBR will be compared against other algorithms on two metrics: the cost and the computation time. The benchmark algorithm is the lower bound given by the multi-path problem obtained by relaxing the single-path constraint. The other three algorithms are:

— **Global Smoothing Algorithm (GSA)**: introduced in [69] for solving non-linear optimization problems with binary variablesto a good approximation of the solution.

— **Ant Colony Optimization (ACO)**: heuristic algorithm proposed in [70]. It belongs to the family of ant colony algorithms, which are known to be efficient for searching an optimal path in a graph [80, 81, 71].

— **Bonmin NLP-based branch-and-bound algorithm**: Bonmin is an open-source software for solving general MINLP (Mixed-Integer Non-Linear Programming) problems [82].

In the experiments, we have used 800 randomly generated instances obtained using 8 standard network topologies collected from the IEEE literature and from the Rocketfuel project [83]. For each network topology, we consider 100 random traffic matrices generated with uniform distributions in such a way that there is positive traffic demand associated to each origin/destination pair. The relative gap (in percentage) to the multi-path lower bound are presented in Table 4.2. PBR and ACO algorithms perform better than GSA, for which relative errors up to 87.54% are obtained on some instances. PBR achieves a lower average relative error with only 1%, whereas ACO has a lower maximum relative error with 14.99%. However, as can be noted from Table 4.3, PBR provides the best tradeoff between quality of the solution and computing times. Over the 800 instances, its worst execution time is only 4.7 seconds, whereas the computing times of GSA and ACO can be as high as 400 and 50 seconds, respectively.

Table 4.2 – Relative gap (%) to the optimal multi-path solution for the M/M/1 cost function.

Topology	**PBR**			**GSA**			**ACO**		
	min	max	avg	min	max	avg	min	max	avg
ABOVENET	$\simeq 0$	1.52	0.72	$\simeq 0$	87.54	5.95	1.94	6.23	3.59
ARPANET	$\simeq 0$	0.50	0.13	$\simeq 0$	77.01	3.44	2.58	7.64	4.82
BHVAC	$\simeq 0$	1.45	0.35	$\simeq 0$	64.58	8.57	3.30	8.54	5.17
EON	$\simeq 0$	1.87	0.82	$\simeq 0$	58.09	3.15	1.71	8.92	3.63
METRO	$\simeq 0$	24.45	2.38	$\simeq 0$	30.89	3.62	0.20	14.99	2.77
NSF	$\simeq 0$	21.06	2.77	0.01	60.14	3.75	0.01	8.77	1.47
PACBELL	$\simeq 0$	1.19	0.20	$\simeq 0$	45.13	5.30	1.44	7.14	4.00
VNSL	$\simeq 0$	4.40	0.64	$\simeq 0$	24.07	3.51	0.34	6.82	2.46
Global	$\simeq 0$	24.45	1.00	$\simeq 0$	87.54	4.53	0.01	14.99	3.26

Table 4.3 – Computing times (seconds) for the 800 problem instances.

	PBR	GSA	ACO	Bonmin
Min	0.03	0.75	4.98	0.6
Max	4.68	400.82	50.95	2638
Average	0.72	27.45	22.40	725

Remark 5. *The results obtained with the standard best-response algorithm are comparable to those presented above for the penalized best-response algorithm. In practice, the results of SBR are sometimes better, and sometimes worse, but the difference is always in the order of a few percents. On the other hand, the computing times of SBR are clearly better since its worst execution time over the* 800 *instances is 1.1 seconds for the M/M/1 cost. This is something expected since the penalty term implies an extra computation with respect to SBR. It is also interesting to note that both versions of the BR algorithm require roughly the same number of iterations.*

Table 4.4 – Relative gap (%) to the optimal multi-path solution for the M/M/1 cost function.

Topology	**penalized BR**			**standard BR**		
	min	max	avg	min	max	avg
ABOVENET	$\simeq 0$	1.52	0.72	$\simeq 0$	1.10	0.46
ARPANET	$\simeq 0$	0.50	0.13	$\simeq 0$	2.99	0.31
BHVAC	$\simeq 0$	1.45	0.35	$\simeq 0$	1.54	0.38
EON	$\simeq 0$	1.87	0.82	$\simeq 0$	4.03	0.52
METRO	$\simeq 0$	24.45	2.38	$\simeq 0$	12.90	1.46
NSF	$\simeq 0$	21.06	2.77	$\simeq 0$	20.85	1.03
PACBELL	$\simeq 0$	1.19	0.20	$\simeq 0$	2.07	0.45
VNSL	$\simeq 0$	4.40	0.64	$\simeq 0$	2.05	0.66
Global	$\simeq 0$	24.45	1.00	$\simeq 0$	20.85	0.67

CHAPTER 5

NON-COOPERATIVE LOAD-BALANCING

A second type of distributed systems is one where the agents are non-cooperative. Each agent minimizes its own and a potentially different cost function unlike in the previous chapter where the costs were designed to arrive at a common goal.

In this chapter, we go back to the load-balancing problem of Chapter 1 except that there are multiple dispatchers and there is no limit on the buffer-size at the servers. Each dispatcher receives requests from its pool of clients and routes each request to a potentially different server. These routing decisions by the dispatchers are taken with the objective of minimizing the cost (for example, the mean sojourn time) incurred by the requests it routes. The dispatchers are thus involved in a non-cooperative game. The present model has two main differences with that of the previous chapter. First, the flows will be allowed to split their traffic over multiple paths. As will be seen, splittability leads to a different (lower) PoA. Second, instead of an arbitrary network topology, here we will limit ourselves to the parallel links topology.

There are numerous applications in communication networks that allow flows to be split. The typical one is that of overlay networks which were in part proposed to offer more flexible routing options than standard one in the Internet. A source encapsulates the original packets into larger packets. Some of these encapsulated packets are sent to intermediate nodes in the overlay with the instruction to forward them to the destination. In standard Internet routing like in the previous chapter, all the packets would follow the same path to the destination. Another example of splittable flows is that of Multi-path TCP which is able to split the data over multiple interfaces.

Contributions and related work

In [BP16], we determine the worst-case scenario for the distributed load-balancing architecture with a finite-number of dispatchers, and to characterize how worse this performance can be. We first show that, for a fixed total incoming traffic to the system, the worst-case performance at the NE occurs when all the dispatchers have the same amount of traffic to be routed, that is, they are involved in a symmetric game. It is known that a symmetric game is a potential game, that is, its equilibrium can be obtained as the solution of a common objective function which is usually much easier to analyse than the NE of game. From this observation, we give a lower bound on the PoA for cost functions of type: (i) $1+x^m$; and (ii) $1/(1-x)^m$.

The distributed load balancing game has been analysed previously by several authors. In [84], similar results were obtained for the $M/M/1$ type delay functions, and the present work is a generalization of that to a larger class of cost functions. In terms of methodology, our work is closely related to that of Orda *et al* [85, 86] and the arguments we use are inspired from their work. However, there are a couple of differences with their work. First, in [85], the authors restrict themselves to the $M/M/1$ delay function, whereas we consider a larger class of cost functions. In particular, we allow for the association of a heterogeneous cost per unit time (holding cost) with each server. The introduction of heterogeneous holding costs can significantly change the PoA for certain cost functions. For example, for the $M/M/1$ delay function and equal holding costs, it was shown in [87] that the PoA is upper bounded by the number of servers whereas for the same model and unequal holding costs, it was shown in [84] that the PoA is of the order of the square root of the number of dispatchers. Second, in [85], it is shown that, for a fixed incoming traffic vector, transferring capacity from a server to another one that has a higher service rate improves the performance at the NE. We look at the complementary problem in which we fix the service capacities, and look at what happens when we transfer traffic from one dispatcher to another.

The computation of PoA for disciplines other than Processor Sharing (or the $M/M/1$ delay function) has also been previously investigated. In [88], the authors compute the PoA of routing policies with memory in server farms with First-Come-First-Served or Processor Sharing service discipline using a lower bound on the cost of the cen-

tralized architecture. The computation of PoA for server farms with SRPT discipline but without memory was given in [89]. The authors compute the PoA on assuming that the system is in heavy-traffic and that all servers are used by all the flows. These results were obtained assuming each request routes itself (*i.e.*, infinite number of dispatchers, one for each request) whereas our results are for a finite number of dispatchers.

5.1 A load-balancing game

The problem formulation is similar to that in the previous chapter except for two main differences: the topology is restricted to that of parallel links, and the flows are splittable. Unlike in the previous chapter, here a node is either a source (dispatcher) or a destination (server). Since there is a clear distinction between the role of nodes, we define some quantities specific to this model.

Denote $\mathscr{C} = \{1,\ldots,K\}$ to be the set of dispatchers and $\mathscr{S} = \{1,\ldots,S\}$ to be the set of servers. Let λ_i be the traffic intensity of jobs received by dispatcher i. These jobs will called jobs of class i. It is assumed that the dispatchers are numbered in order of their traffic intensities, *i.e.*, $\lambda_i \le \lambda_j$ for $i \le j$. Moreover, it will also be assumed that the vector $\boldsymbol{\lambda}$ of traffic intensities belongs to the following set:

$$\Lambda = \left\{ \boldsymbol{\lambda} \in \mathbb{R}^K \ : \ \sum_{i\in\mathscr{C}} \lambda_i = \overline{\lambda} \right\},$$

where $\bar{\lambda}$ denotes the total incoming traffic intensity

Server $j \in \mathscr{S}$ has capacity r_j and a holding cost c_j per unit time is incurred for each job sent to this server. Let $u_j = \frac{c_j}{r_j}$ denote the ratio of the holding cost to the capacity for server j. By convention, $u_{S+1} = \infty$. Instead of implicitly defining these quantities within the link delay function, they are explicitly defined since we will show some properties of the Nash equilibrium based on u_j. It is assumed that servers are numbered in the order of increasing cost per unit capacity, i.e., if $m \le n$, then $u_m \le u_n$, so that the first servers are more attractive than the last ones. Let $\mathbf{r} = (r_j)_{j\in\mathscr{S}}$ and $\mathbf{c} = (c_j)_{j\in\mathscr{S}}$, and let $\overline{r} = \sum_{n\in\mathscr{S}} r_n$ denote the total capacity of the system.

Let $\mathbf{x}_i = (x_{i,j})_{j\in\mathscr{S}}$ denote the routing strategy of dispatcher i, with $x_{i,j}$ being the

amount of traffic it sends towards server j. Let

$$\mathscr{X}_i = \left\{ \mathbf{x}_i \in \mathbb{R}^S \,:\, 0 \le x_{i,j} \le r_j \,, \forall j \in \mathscr{S}; \sum_{j \in \mathscr{S}} x_{i,j} = \lambda_i \right\}$$

denote the set of feasible routing strategies for dispatcher i. The vector $\mathbf{x} = (\mathbf{x}_i)_{i \in \mathscr{C}}$ will be called a multi-strategy. The multi-strategies belong to the product strategy space $\mathscr{X} = \bigotimes_{i \in \mathscr{C}} \mathscr{X}_i$. The model is illustrated in Figure 5.1.

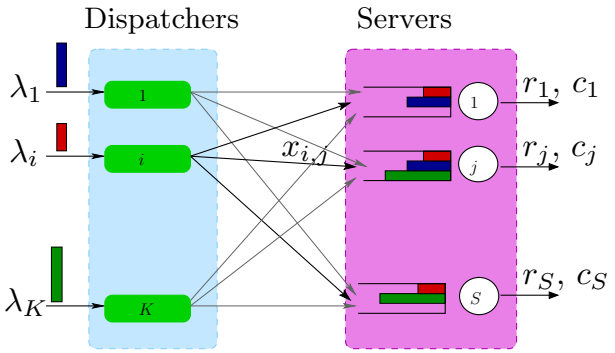


Figure 5.1 – Distributed load balancing.

Dispatcher i seeks to minimize its cost function which is denoted by $f_i(\mathbf{x})$. This optimization problem can be formulated as follows:

$$\underset{\mathbf{x} \in \mathscr{X}_i}{\text{minimize}}\; f_i(\mathbf{x}) = \sum_{j \in \mathscr{S}} u_j x_{i,j} \ell_j(\rho_j)$$

where $\rho_j = \sum_{i \in \mathscr{C}} x_{i,j} / r_j$ is the load on server j.

The congestion function ℓ has to have certain properties for the existence and uniqueness of the NE. For some $\beta > 0$, it will be assumed that the congestion function $\ell : [0, \beta) \to [1, \infty)$ is a continuous, strictly increasing and convex function. Further ℓ is continuously differentiable and its second derivative exists. We also need $\ell(0) = 1$ and $\lim_{\rho \to \beta^-} \ell(\rho) = +\infty$.

Examples of functions with $\beta = \infty$ include polynomials as well as functions of the type $\exp(\rho)$. When $\beta < \infty$, it will be assumed that $\bar{\lambda} < \bar{r}$, which is the necessary and sufficient condition to guarantee the stability of the system. Examples of such func-

tions are those from queueing theory, e.g. the M/G/1/PS delay function $\phi(\rho) = \frac{1}{1-\rho}$ or the M/G/1/FCFS delay function $\phi(\rho) = 1 + \rho\,\frac{1+c_b^2}{2(1-\rho)}$, where c_b^2 denotes the squared coefficient of variation of the job sizes. This class of functions also contains the delay function of the $M/Pareto/1/SRPT$ in heavy-traffic, which is given by $\frac{1}{(1-\rho)^m}$, where m depends on the shape parameter of the Pareto distribution [89].

For functions satisfying these assumptions, the existence and uniqueness of NE was shown in [86]. The PoA is then defined as

$$PoA(K) = \sup_{\boldsymbol{\lambda},\mathbf{r},\mathbf{c}} \frac{D_K(\boldsymbol{\lambda},\mathbf{r},\mathbf{c})}{D_1(\bar{\lambda},\mathbf{r},\mathbf{c})},$$

where $D_K(\boldsymbol{\lambda},\mathbf{r},\mathbf{c})$ is the total cost of the NE of the game with K dispatchers and is given by $D_K(\boldsymbol{\lambda},\mathbf{r},\mathbf{c}) = \sum_{j\in\mathscr{S}} c_j\,\rho_j\,\phi(\rho_j)$.

Since the centralized model can be thought of as a game with just one dispatcher routing all the traffic, its optimum cost can be defined in terms of the function D as $D_1(\bar{\lambda},\mathbf{r},\mathbf{c})$.

5.2 Impact of the traffic heterogeneity

Our main result states that that the global cost $D_K(\boldsymbol{\lambda},\mathbf{r},\mathbf{c})$ achieves its maximum when $\boldsymbol{\lambda}$ is the symmetric vector $\boldsymbol{\lambda}^{=} = \left(\frac{\bar{\lambda}}{K},\ldots,\frac{\bar{\lambda}}{K}\right)$.

Theorem 11.

$$\sup_{\boldsymbol{\lambda},\mathbf{r},\mathbf{c}} D_K(\boldsymbol{\lambda},\mathbf{r},\mathbf{c}) = \sup_{\mathbf{r},\mathbf{c}} D_K(\boldsymbol{\lambda}^{=},\mathbf{r},\mathbf{c}).$$

The proof of this theorem is similar to the proof of the corresponding theorem for the $M/M/1$ delay functions in [84]. It amounts to proving that, starting from an arbitrary traffic vector $\boldsymbol{\lambda}$, the symmetric traffic vector $\boldsymbol{\lambda}^{=}$ can be reached by a sequence $\{\boldsymbol{\lambda}^n\}_{n\geq 0}$ such that $\boldsymbol{\lambda}^0 = \boldsymbol{\lambda}$ and $D_K(\boldsymbol{\lambda}^{n+1}) \geq D_K(\boldsymbol{\lambda}^n)$. Such a sequence is obtained by considering a certain transformation that transfers traffic from the most loaded dispatchers to the least loaded ones, thus reducing the heterogeneity of the traffic vector. It then remains to be proven that the social cost, D_K increases under this trans-

formation as well as that this sequence of transformation converges in finite number of steps. The convergence follows from Proposition 7 of [84] since their proof is not specific to M/M/1 delay function. For the monontonicity, again the approach is similar to that in [84] but this time it needs to be generalized to the considered set of cost functions. It is basically enough to prove that the social cost D_K is non-decreasing under the transformation when the set of servers used under the original and the transformed traffic vectors coincide provided the Nash mapping associating to a vector $\boldsymbol{\lambda}$ the strategy profile $\mathscr{N}(\boldsymbol{\lambda}) \in \mathscr{X}$ is continuous. This can be proven by a straightforward generalization of Theorem 3 in [84] to general cost function ϕ (see [90] for details).

Properties of the NE

Certain properties of NE help in establishing the proof. They are also interesting in themselves, *e.g.*, they show which classes use which servers.

Let $\mathscr{S}_i$ be the set of servers used by class i and let $\mathscr{C}_j$ be the set of classes that use server j. Let μ_i be the marginal cost of class i on servers to which it sends jobs to at the NE. It can be shown that there is a monotonicity among classes in their use of servers: a class with a higher demand uses more of each and every server.

Proposition 7. *The following statements are equivalent:*

1. $\mathscr{S}_i \subseteq \mathscr{S}_k$.
2. $x_{i,j} < x_{k,j}$, $\forall\, j \in \mathscr{S}_k$.
3. $\lambda_i < \lambda_k$.

A class with less total traffic uses a subset of servers used by a class with more total traffic. Further, the traffic sent to each server is also smaller. As a consequence, the set of classes using server j has the following structure: $\mathscr{C}_j = \{K - N_j + 1, \ldots, K\}$, where N_j is the number of classes sending a positive flow to server j. The following proposition gives analogous properties for the servers.

Proposition 8. *The following statements are equivalent:*

1. $\mathscr{C}_m \subseteq \mathscr{C}_n$.

2. $\frac{u_m x_{i,m}}{r_m}\phi'(\rho_m) < \frac{u_n x_{i,n}}{r_n}\phi'(\rho_n),\ \forall i \in \mathscr{C}_n$..
3. $u_n < u_m$.

It follows that the set $\mathscr{S}_i$ has the following structure: $\mathscr{S}_i = \{1,\dots,S_i\}$, where $S_i = |\mathscr{S}_i|$ is the number of servers used by class i.

One last property pertains to the marginal private costs of the servers It measures the change in social cost D_K that arises when the offered traffic of this server changes by an infinitesimal amount, and is given by

$$\frac{\partial D_K}{\partial y_j}(\mathbf{x}) = u_j\left[\phi(\rho_j) + \frac{y_j}{r_j}\phi'(\rho_j)\right]. \tag{5.1}$$

The lower the cost per unit capacity of a server, greater is its marginal social cost at the NE.

Lemma 1.

$$\frac{\partial D_K}{\partial y_j}(\mathbf{x}) \geq \frac{\partial D_K}{\partial y_{j+1}}(\mathbf{x}), \forall j,$$

with strict inequality if $\mathscr{C}_j \setminus \mathscr{C}_{j+1} \neq \emptyset$.

Due to the convexity of the social cost, Lemma 1 implies that the social cost at the NE can only be decreased by reducing the load of the servers with the lowest cost per unit capacity. The transformation we shall now define has the opposite effect, *i.e.*, it will increase the load on the servers with lower u_j.

The transformation and its impact on server loads

The transformation increases the traffic of classes with the smallest amount of traffic and decreases correspondingly the traffic of classes with the largest amount of traffic such that the total traffic remains the same. The transformation is stopped just before the set of servers used at the transformation can change from that at the initial traffic vector.

In order to determine the impact of the transformation $\boldsymbol{\lambda} \to \hat{\boldsymbol{\lambda}}$ on the global cost, we need to compare the server loads under the equilibria $\mathbf{x}$ and $\hat{\mathbf{x}}$ associated to $\boldsymbol{\lambda}$ and

$\hat{\boldsymbol{\lambda}}$, respectively. To this end, let us define the sets $\mathscr{S}^+$ and $\mathscr{S}^-$ as follows:

$$\mathscr{S}^+ = \{j \in \mathscr{S} \,:\, \hat{y}_j > y_j\} \quad \text{and} \quad \mathscr{S}^- = \mathscr{S} \setminus \mathscr{S}^+,$$

i.e., $\mathscr{S}^+$ is the set of servers whose load increases under the transformation while $\mathscr{S}^-$ is the set of servers whose load is non-increasing under the transformation.

This transformation impacts the server loads in the following way. First, if there exists at least one server whose load increases under the transformation, then the load of each and every server used by class 1 increases.

Proposition 9. *If $\mathscr{S}^+ \neq \emptyset$ then $\mathscr{S}_1 \subset \mathscr{S}^+$.*

Second, the load of all servers is non-increasing under the transformation if and only if all traffic classes use the same set of servers. Further, if at equilibria $\mathbf{x}$ and $\hat{\mathbf{x}}$ all classes use the same set of servers, then the server loads are constant under the transformation.

Proposition 10. *The following statements are equivalent :*

1. $\mathscr{S}^+ = \emptyset$.
2. $\mathscr{S}_1 = \mathscr{S}_K$.
3. $y_j = \hat{y}_j, \forall\, j \in \mathscr{S}$.

Finally, the transformation induces a monotonic partition of servers: there exists a threshold $J < S$ such that for all servers $j > J$ the load is non-increasing under the transformation.

Proposition 11. *For all $j \in \mathscr{S}$, if $j \in \mathscr{S}^-$ then $j+1 \in \mathscr{S}^-$.*

Impact on the social cost of reduction in traffic heterogeneity

The above results can be used to compare the costs $D(\boldsymbol{\lambda})$ and $D(\hat{\boldsymbol{\lambda}})$ before and after the transformation. We first consider the case where not all classes use the same set of servers at the equilibrium $\mathbf{x}$. Here, the transformation will strictly increase the cost.

Theorem 12. $D(\boldsymbol{\lambda}) < D(\hat{\boldsymbol{\lambda}}) \iff \mathscr{S}_1 \subsetneq \mathscr{S}_K$.

The next results proves that the cost is constant under the transformation if all classes use the same set of servers.

Theorem 13. $D(\boldsymbol{\lambda}) = D(\hat{\boldsymbol{\lambda}}) \iff \mathscr{S}_1 = \mathscr{S}_K$.

Theorem 11 now follows from Theorems 12 and 13. Further, it can can also be shown that the PoA is a non-decreasing function of the number of dispatchers.

Corollary 7.

$$PoA(K) \leq PoA(K+1), \quad \forall K \geq 1. \tag{5.2}$$

Analysis of the Symmetric Game

Theorem 11 implies that the PoA can be obtained from the worst-case analysis of the symmetric game. It is well known that the symmetric non-cooperative routing game is a potential game, *i.e.*, the equilibrium flows are the global minima of a standard convex optimization problem (see *e.g.*, Theorem 4.1 in [91]). Formally,

Proposition 12. *If the vector* $\boldsymbol{\rho}$ *is global optimum of the following convex optimization problem*

$$\begin{aligned} \underset{\boldsymbol{\rho}}{\text{minimize}} \quad & \sum_{j \in \mathscr{S}} c_j \rho_j \phi(\rho_j) + (K-1) \int_0^{\rho_j} c_j \phi(z) dz \\ \text{s.t.} \quad & \\ & \textstyle\sum_{j \in \mathscr{S}} r_j \rho_j = \bar{\lambda}, \\ & 0 \leq \rho_j < 1, \ \forall j \in \mathscr{S}, \end{aligned}$$

then the multi-strategy $\mathbf{x}$ *such that* $x_{i,j} = r_j \frac{\rho_j}{K}$, $\forall i \in \mathscr{C}$, $\forall j \in \mathscr{S}$, *is the NE of the symmetric game.*

Note that when $K = 1$, the above problem reduces to the global optimization problem solved by the centralized scheme. When $K \to \infty$, the equivalent problem states the common function that is optimized jointly by an infinite number of players and is characteristic of the Wardrop equilibrium.

5.3 Lower bound on Price of Anarchy

We now give lower bounds on the PoA for the functions of type (i) $\phi(x) = \frac{1}{(1-x)^m}$, and (ii) $\phi(x) = 1 + x^m$. For this, we construct an example with two servers and symmetric dispatchers. From the analysis of this symmetric game, it can be shown that the ratio D_K/D_1 attains the claimed lower bound.

Proposition 13. *For* $\phi(x) = \frac{1}{(1-x)^m}$,

$$PoA(K) \geq \frac{K}{\left(K^{1/(m+1)} + mK^{1/(m+1)} - m\right)}. \tag{5.3}$$

For functions of type $\phi(x) = 1 + x^m$, the result is in the form of a conjecture although we provide strong numerical evidence to support it.

Conjecture 2. *For* $\phi(x) = 1 + x^m$ *and* $m \geq 1$,

$$PoA(K) \geq \frac{(1+m/K)^{-1}}{(1+m/K)^{-1}\left(\frac{1+m/K}{1+m}\right)^{\frac{m+1}{m}} + m^{-1}\log\left(\frac{1+m}{1+m/K}\right)}. \tag{5.4}$$

For this example, it can be shown that

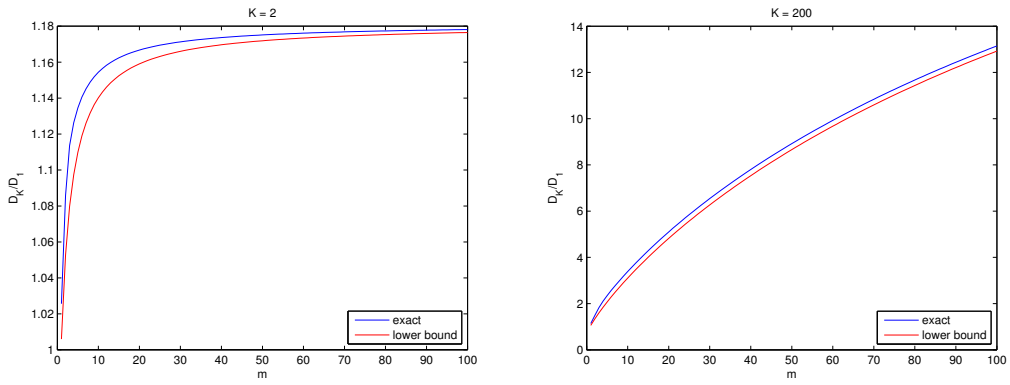
$$\frac{D_K}{D_1} = \frac{(1+m/K)^{-1}}{(1+m/K)^{-1}(1-\rho_2)^{m+1} + \rho_2(1+\rho_2^m)}, \tag{5.5}$$

where ρ_2 is the solution of

$$\left(\frac{1+m/K}{1+m}\right)^{1/m}(1+(1+m)\rho_2^m)^{1/m} + \rho_2 = 1.$$

We now claim that the denominator of D_K/D_1 is upper bounded by $(1+m/K)^{-1}\left(\frac{1+m/K}{1+m}\right)^{\frac{m+1}{m}} + m^{-1}\log\left(\frac{1+m}{1+m/K}\right)$, which leads to the conjecture. The numerical evidence for this is shown in Figures 5.2a and 5.2b where the exact value of D_K/D_1 is compared with the lower bound for various values of m and K.

Remark 6. *1. The lower bounds obtained above are independent of* $\mathbf{r}$ *and* $\mathbf{c}$.



(a) $K = 2$ (b) $K = 200$

Figure 5.2 – Comparison of D_K/D_1 and its lower bound for $m = 1, 2, ..., 100$.

2. *As $K \to \infty$, the lower bound of (5.4) tends to a constant, unlike the lower bound (5.3) which tends to infinity. Thus, for delay function of the type of $M/Pareto/1/SRPT$ queues, the PoA can be unbounded when each job minimizes its own mean delay.*
3. *Moreover, for $K = \infty$, the lower bound (5.4) is of the order of $m/\log(m)$, which matches the PoA obtained by Roughgarden [92] for polynomial functions for the Wardrop equilibrium.*
4. *For $m = 1$ the delay function of $M/Pareto/1/SRPT$ is the same as that of the $M/G/1/PS$. From the lower bound formula given above, we retrieve the lower bound in [84].*

Although we do not have the upper bounds for PoA as in [84], we conjecture that the lower bounds constructed using the above method give the right order of the PoA, just as was proved in [84] for the case of $M/M/1$ delay functions. One of the main difficulties in proving this conjecture is that, except for the $M/M/1$ case, a closed-form solution of the optimization problem stated in Proposition 12 cannot be obtained. This turns out to be a major obstacle for the derivation of upper bounds on the PoA.

PART III

Perspectives

PERSPECTIVES

The availability of large amounts of data and cheap computing resources is driving research in the direction of finding solutions that require fewer hypothesis and that are more appropriate for the target application. Recall that in the stochastic approach it not always easy to determine the optimal policy (for routing, *e.g.*) unless some assumptions are imposed which are not always practical. To obtain policies in more general settings, learning-based methods have been investigated widely in recent years. Take the load-balancing application of Chapters 1 and 5. Data collection has become the norm in digital environments and data-centers can obtain large amounts of data on the demand profile as well as on the profile of the service requirements of requests. This data need not fit the assumptions for the policies that are easily analyzable with stochastic models. In machine learning, the operator does not need to worry about meeting assumptions. It runs learning algorithms (say reinforcement learning) on the dataset and explores various policies with the goal of finding the best one suitable for the given dataset. One can thereby obtain solutions that are tailored for individual data-centers and the profiles of requests that is treats.

This approach works well when there is large amount of data and computing power available. Even then the convergence can be slow for large networks since a correspondingly large state-space has to be explored. A research direction that I intend to pursue involves methods to improve the convergence of these algorithms. Usually these algorithms start with a random initialization, that is with a policy that is arbitrary. This initialization may be far away from the optimal policy and may force the learning algorithm to explore much more before converging. A possible way to improve upon this is to warm-start the network with a policy that is known to good in some sense. In the load-balancing, example, this could be JSQ or the optimal reversible policy. This way, the leaning algorithm starts from a decent policy and will only improve it. In order to save energy, one need not wait for the optimum to be reached and can stop earlier knowing it has at least found something better than an already decent policy. In case of warm-start, the stopping decision can hopefully be made much earlier than for random initialization. A second possible method for im-

proving convergence is to restrict the learning algorithm to policies that are in a certain class. For example, the reversible policies, or of weighted JSQ type for which one has to learn the optimal weights. Here, by restricting the class of policies, a smaller and more compact network may be sufficient and less exploration may be required. In both these methods, we are relying upon policies with certain desirable properties which are obtained by theoretical analysis (either stochastic or worst-case). This research direction can be seen as a combination of the analytical approach and the data-based learning approach in order to obtain faster and improved algorithms tailored for applications. It can be investigated for both centralized as well as in the distributed architectures. For example, for the single-path routing in Chapter 4, the learning could be performed by agents that are distributed across the network. In addition to the question of convergence, one could also investigate whether the tweaked objective function works well in practical or other specific penalties will have to be designed in to obtain good approximations of the global optimum.

A second research direction is also based on availability of data and is motivated by the observation they can be used to provide predictions which can improve some of the decision-making. In previous works, others [93] and us [94] have explored this possibility for scheduling in vehicular networks where the decisions are which vehicles to send data to and the associated power levels. With the availability of SNR maps that give the channel conditions according to the spatial position, one can predict the conditions a vehicle can expect to be in the short-term. If they are expected to be unfavorable, then it may help to give additional data rate to the vehicle while the conditions are better. By superposing the predicted trajectories of vehicles and the SNR maps, one can improve upon the widely-used proportional-fair allocation policy which only relies upon current and past conditions for making decisions. In this context, we intend to investigate the improvement that can be expected from incorporating information on the future into current decision making. This information could be channel conditions as in wireless networks or could be the profile of tasks in the load-balancing example. However, future information need not be exact and may contain errors. The algorithms will have account for these and be robust inaccurate predictions. Also, gathering future information may incur a cost which may induce a tradeoff between the amount of improvement and the acquisition cost.

Again the algorithms (or the class of algorithms) identified from this approach can potentially serve for improved convergence in the first research direction. We note that this avenue of integrating future information has also been explored in the context of online approximation algorithms [95]. My focus will be more on the stochastic models based approach.

Finally, I mention two problems related to the game-theoretic analysis of Chapter 5. The first is a generalization of the worst-case scenario to a network with a given traffic matrix, the goal being to arrive at the PoA in a network setting. The second problem is the convergence of the best-response algorithm which is still an open problem for this game. We proposed an approach based on joint-spectral radius (JSR) of the set of Jacobian matrices of the best-response operator [96]. We showed that the Jacobian matrices for the load-balancing game have a certain structure. Unfortunately, JSR is typically not an easy quantity to compute even with the special structure of our set of operators. This is being currently explored for some simplified cost functions and scenarios in order to obtain some insights before going to the general case.

PUBLICATIONS RELATED TO THE THESIS

[BP16] O. Brun and B.J. Prabhu. Worst-case analysis of non-cooperative load balancing. *Annals of Operations Research*, 239(2):471–495, 2016.

[BPV17] O. Brun, B.J. Prabhu, and J. Vallet. A penalized best-response algorithm for nonlinear single-path routing problems. *Networks*, 69(1):52–66, 2017.

[FJP14] M. Feuillet, M. Jonckheere, and B.J. Prabhu. Bandwidth sharing networks with multiscale traffic. *Stochastic Systems*, 4(2):449–478, 2014.

[JP18] M. Jonckheere and B.J. Prabhu. Asymptotics of insensitive load balancing and blocking phases. *Queueing Syst.*, 88(3-4):243–278, 2018.

[NQPR23] R. Núñez-Queija, B.J. Prabhu, and J.A.C. Resing. Markovian queues with poisson control. *Indagationes Mathematicae*, 2023.

BIBLIOGRAPHY

[1] David P. Williamson and David B. Shmoys. *The Design of Approximation Algorithms*. Cambridge University Press, 2011.

[2] Niv Buchbinder and Joseph (Seffi) Naor. The design of competitive online algorithms via a primal–dual approach. *Foundations and Trends® in Theoretical Computer Science*, 3(2–3):93–263, 2009.

[3] Mor Harchol-Balter. *Performance Modeling and Design of Computer Systems: Queueing Theory in Action*. Cambridge University Press, USA, 1st edition, 2013.

[4] Eugene A. Feinberg and Adam Shwartz, editors. *Handbook of Markov Decision Processes*. Springer, 2001.

[5] Philippe Robert. *Stochastic Networks and Queues*. Springer, 2003.

[6] Ward Whitt. *Stochastic-Process Limits*. Springer, 2002.

[7] Amir Dembo and Ofer Zeitouni. *Large Deviations Techniques and Applications*. Springer, 2009.

[8] Michel Benaïm and Jean-Yves Le Boudec. A class of mean field interaction models for computer and communication systems. *Performance Evaluation*, 65(11):823–838, 2008. Performance Evaluation Methodologies and Tools: Selected Papers from ValueTools 2007.

[9] Sean Meyn. *Control Techniques for Complex Networks*. Cambridge University Press, 2007.

[10] J. Michael Harrison. *Brownian Models of Performance and Control*. Cambridge University Press, 2013.

[11] Nicolas Gast, Bruno Gaujal, and Jean-Yves Le Boudec. Mean field for markov decision processes: From discrete to continuous optimization. *IEEE Transactions on Automatic Control*, 57(9):2266–2280, 2012.

[12] Y. Narahari. *Game Theory and Mechanism Design*. World Scientific/Indian Inst. of Science, India, 2014.

[13] Zhu Han, Dusit Niyato, Walid Saad, and Tamer Başar. *Game Theory for Next Generation Wireless and Communication Networks: Modeling, Analysis, and Design*. Cambridge University Press, 2019.

[14] David Easley and Jon Kleinberg. *Networks, Crowds, and Markets: Reasoning about a Highly Connected World*. Cambridge University Press, 2010.

[15] N. Nisan, T. Roughgarden, E. Tardos, and V. Vazirani, editors. *Algorithmic Game Theory*. Cambridge University Press, 2007.

[16] A. Hordijk and G. Koole. On the optimality of the generalized shortest queue policy. *Probability in the Engineering and Informational Sciences*, 4(4):477–487, 1990.

[17] P. D. Sparaggis, D. Towsley, and C. G. Cassandras. Extremal properties of the shortest/longest non-full queue policies in finite-capacity systems with state-dependent service rates. *Journal of Applied Probability*, 30(1):223–236, 1993.

[18] Pravin K. Johri. Optimality of the shortest line discipline with state-dependent service rates. *European Journal of Operational Research*, 41(2):157–161, 1989.

[19] Ward Whitt. Deciding which queue to join: Some counterexamples. *Operations Research*, 34(1):55–62, 1986.

[20] C. Knessl and H. Yao. On the finite capacity shortest queue problem. *Progress in Applied Mathematics*, 2(1):1–34, 2011.

[21] P. Eschenfeldt and D. Gamarnik. Join the shortest queue with many servers. the heavy traffic asymptotics, 2015.

[22] D. L. Jagerman. Some properties of the Erlang loss function. *Bell System Technical Journal*, 53(3):525–551, 1974.

[23] Shlomo Halfin and Ward Whitt. Heavy-traffic limits for queues with many exponential servers. *Operations Research*, 29(3):567–588, 1981.

[24] F. Kelly. *Reversibility and Stochastic Networks*. Wiley, 1979.

[25] T. Bonald and A. Proutìere. Insensitivity in processor-sharing networks. *Performance Evaluation*, 49:193–209, 2002.

[26] T. Bonald, M. Jonckheere, and A. Proutière. Insensitive load balancing. *SIGMETRICS Perform. Eval. Rev.*, 32(1):367–377, 2004.

[27] M. Bramson, Y. Lu, and B. Prabakhar. Asymptotic independence of queues under randomized load balancing. *Queueing Syst*, 71:247–292, 2012.

[28] W. Whitt. Heavy traffic approximations for service systems with blocking. *Bell System Technical Journal*, 63(4):689–708, 1984.

[29] A. J. E. M. Janssen, J. S. H. Van Leeuwaarden, and B. Zwart. Gaussian expansions and bounds for the poisson distribution applied to the erlang b formula. *Advances in Applied Probability*, 40(1):122–143, 2008.

[30] A. J. E. M. Janssen, J. S. H. van Leeuwaarden, and B. Zwart. Refining square-root safety staffing by expanding erlang c. *Operations Research*, 59(6):1512–1522, 2011.

[31] R. Atar. A diffusion regime with nondegenerate slowdown. *Operations Research*, 60(2):490–500, 2012.

[32] S. Maman. Uncertainty in the demand for service: The case of call centers and emergency departments. Master's thesis, Technion, April 2009.

[33] Yi Lu, Qiaomin Xie, Gabriel Kliot, Alan Geller, James R. Larus, and Albert Greenberg. Join-idle-queue: A novel load balancing algorithm for dynamically scalable web services. *Perform. Eval.*, 68(11):1056–1071, November 2011.

[34] M. Mitzenmacher. The power of two choices in randomized load balancing. *Ph.D. Thesis*, 1996.

[35] L. Massoulié and J. Roberts. Bandwidth sharing: objectives and algorithms. *IEEE/ACM Trans. Netw.*, 10(3):320–328, 2002.

[36] T. Bonald and A. Proutière. Insensitive bandwidth sharing in data networks. *Queueing Syst. Theory Appl.*, 44(1):69–100, 2003.

[37] H. C. Gromoll and R. J. Williams. Fluid limits for networks with bandwidth sharing and general document size distributions. *Annals of Applied Probability*, 19:243, 2009.

[38] T. Bonald, L. Massoulié, A. Proutière, and J. Virtamo. A queueing analysis of max-min fairness, proportional fairness and balanced fairness. *Queueing Syst. Theory Appl.*, 53(1-2):65–84, 2006.

[39] T. Bonald and A. Proutière. On performance bounds for the integration of elastic and adaptive streaming flows. In *SIGMETRICS*, pages 235–245, 2004.

[40] J. Michael Harrison, Chinmoy Mandayam, Devavrat Shah, and Yang Yang. Resource sharing networks: Overview and an open problem. *Stochastic Systems*, 4(2):524 – 555, 2014.

[41] I.M. Verloop. *Scheduling in stochastic resource-sharing system*. PhD thesis, Eindhoven University of Technology, 2009.

[42] M. Deshpande, A. Amit, M. Chang, N. Venkatasubramanian, and S. Mehrotra. Flashback: A peer-to-peer web server for flash crowds. *International Conference on Distributed Computing Systems*, page 15, 2007.

[43] T. Peng, C. Lecki, and K. Ramamohanarao. Survey of network-based defense mechanisms countering the DoS and DDoS problems. *ACM Comput. Surv.*, 39(1), 2007.

[44] A. Stavrou, D. Rubenstein, and S. Sahu. A lightweight, robust p2p system to handle flash crowds. *Selected Areas in Communications, IEEE Journal on*, 22(1):6–17, 2004.

[45] S. Kandula, D. Katabi, M. Jacob, and A .W. Berger. Botz-4-Sale: Surviving Organized DDoS Attacks That Mimic Flash Crowds. In *2nd Symposium on Networked Systems Design and Implementation (NSDI)*, Boston, MA, May 2005.

[46] R. Núñez-Queija, J. van den Berg, and M. Mandjes. Performance evaluation of strategies for integration of elastic and stream traffic. In *ITC 16*, pages 1039–1050, 1999.

[47] Sunil Kumar and Laurent Massoulié. Integrating streaming and file-transfer internet traffic: fluid and diffusion approximations. *Queueing Syst.*, 55(4):195–205, 2007.

[48] F. Kelly. Blocking probabilities in large circuit-switched networks. *Adv. Appl. Probab.*, 18:473–505, 1986.

[49] Philippe Robert. *Stochastic Networks and Queues*. Stochastic Modelling and Applied Probability Series. Springer-Verlag, New York, 2003. xvii+398 pp.

[50] J. G. Dai. On positive harris recurrence of multiclass queueing networks: A unified approach via fluid limit models. *Annals of Applied Probability*, 5:49–77, 1995.

[51] R. W. R. Darling and J. R. Norris. Differential equation approximations for markov chains. *Probability Surveys*, 5:37, 2008.

[52] S. Meyn. *Control techniques for complex networks*. Cambridge University Press, 2008.

[53] J. Mo and J. Walrand. Fair end-to-end window-based congestion control. *IEEE/ACM Trans. on Netw.*, 8(5):556–567, 2000.

[54] T. Bonald and L. Massoulié. Impact of fairness on Internet performance. In *Proceedings of ACM SIGMETRICS/Performance*, pages 82–91, 2001.

[55] N. Bansal, T. Kimbrel, and K. Pruhs. Speed scaling to manage energy and temperature. *J. ACM*, 54(1):3:1–3:39, March 2007.

[56] L.L.H. Andrew, Minghong M. Lin, and A. Wierman. Optimality, fairness, and robustness in speed scaling designs. In *Proceedings of the ACM SIGMETRICS'10*, pages 37–48, New York, USA, 2010. ACM.

[57] J.W. Cohen. *The Single Server Queue*. Elsevier Science, 1982.

[58] G. Latouche and V. Ramaswami. *Introduction to Matrix Analytic Methods in Stochastic Modeling*. ASA-SIAM Series on Statistics and Applied Probability. SIAM, 1999.

[59] Attila Lovas and Miklós Rásonyi. Markov chains in random environment with applications in queuing theory and machine learning. *Stochastic Processes and their Applications*, 137:294–326, 2021.

[60] Yiran Liu, Harsha Honnappa, Samy Tindel, and Nung Kwan Yip. Infinite server queues in a random fast oscillatory environment. *Queueing Systems*, 98(1):145–179, Jun 2021.

[61] S.K. Cheung, R.J. Boucherie, and R. Núñez-Queija. Quasi-stationary analysis for queues with temporary overload. In *2010 22nd International Teletraffic Congress (ITC 22)*, pages 1–8, 2010.

[62] Guy Latouche and Marcel F. Neuts. Efficient algorithmic solutions to exponential tandem queues with blocking. *SIAM J. Algebra. Discr.*, 1:93–106, 1980.

[63] M. Pióro and D. Medhi. *Routing, flow, and capacity design in communication and computer networks*. Elsevier, 2004.

[64] B. Davie and Y. Rekhter. *MPLS Technology and Applications*. Morgan Kaufmann, 2000.

[65] M. Jünger, Th.M. Liebling, D. Naddef, G.L. Nemhauser, W.R. Pulleyblank, G. Reinelt, G. Rinaldi, and L.A. (Eds.) Wolsey, editors. *50 years of Integer Programming 1958–2008*. Springer Berlin, 2010.

[66] P. Belotti, C. Kirches, S. Leyffer, J. Linderoth, J. Luedtke, and A. Mahajan. Mixed-integer nonlinear optimization. *Acta Numerica*, 22:1–131, 2013.

[67] E. Cela. *The Quadratic Assignment Problem: Theory and Algorithms*. Kluwer Academic Publishers, 1998.

[68] Panos M. Pardalos. Continuous approaches to discrete optimization problems. In G. Di Pillo and F. Giannessi, editors, *Nonlinear Optimization and Applications*, pages 313–325. Springer US, 1996.

[69] Walter Murray and Kien-Ming Ng. An algorithm for nonlinear optimization problems with binary variables. *Computational Optimization and Applications*, 47(2):257–288, 2010.

[70] J.-M. Garcia, A. Rachdi, and O. Brun. Optimal LSP placement with QoS constraints in Diffserv/MPLS networks. In J. Charzinski, R. Lehnert, and P. Tran-Gia, editors, *Proc 18th International Teletraffic Congress (ITC-18)*, volume 5 of *Teletraffic Science and Engineering*, pages 11 – 20. Elsevier, 2003.

[71] Ruud Schoonderwoerd, Owen Holl, Janet Bruten, and Leon Rothkrantz. Ant-based load balancing in telecommunications networks. *Adaptive Behavior*, 5:169–207, 1996.

[72] Drew Fudenberg and Jean Tirole. *Game Theory*. MIT Press, Cambridge, MA, 1991.

[73] A. Al Sheikh, O. Brun, P.E. Hladik, and B.J. Prabhu. A best-response algorithm for multiprocessor periodic scheduling. In *23rd Euromicro Conference on Real-Time Systems (ECRTS)*, pages 228–237, July 2011.

[74] P. Coucheney. *Auto-optimisation des réseaux sans fil : Une approche par la théorie des jeux*. PhD thesis, Université de Grenoble, 2006.

[75] A. Fabrikant, C. Papadimitriou, and K. Talwar. The complexity of pure Nash equilibria. In *Proceedings of the Thirty-sixth Annual ACM Symposium on Theory of Computing*, STOC '04, pages 604–612, New York, NY, 2004. ACM.

[76] S. Durand and B. Gaujal. Complexity and optimality of the best-response algorithm in random potential games. Research Report RR-8925, Inria - Research Centre Grenoble – Rhône-Alpes ; Grenoble 1 UGA - Université Grenoble Alpe, June 2016.

[77] Baruch Awerbuch, Yossi Azar, and Amir Epstein. The price of routing unsplittable flow. In *Proc Thirty-seventh Annual ACM Symposium on Theory of Computing (STOC '05)*, pages 57–66, Baltimore, MD, 2005. ACM.

[78] K. Bhawalkar, M. Gairing, and T. Roughgarden. Weighted congestion games: The price of anarchy, universal worst-case examples, and tightness. *ACM Trans. Economics and Comput.*, 2(4):14:1–14:23, 2014.

[79] S. Aland, D. Dumrauf, M. Gairing, B. Monien, and F. Schoppmann. Exact price of anarchy for polynomial congestion games. *SIAM Journal on Computing*, 40:1211–1233, 2011.

[80] M. Dorigo. *Optimization, learning and natural algorithms*. PhD thesis, Politecnico di Milano, Italy, 1992.

[81] M. Dorigo, V. Maniezzo, and A. Colorni. Ant system: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics*, 26(1):29–41, 1996.

[82] P. Bonami, L.T. Biegler, A.R. Conn, G. Cornuejols, I.E. Grossmann, C.D. Laird, J. Lee, A. Lodi, F. Margot, N.Sawaya, and A. Waechter. An algorithmic framework for convex mixed integer nonlinear programs. *Discrete Optimization*, 5(2):186–204, 2008.

[83] N. Spring, R. Mahajan, D. Wetherall, and T. Anderson. Measuring ISP topologies with Rocketfuel. *IEEE/ACM Transactions on Networking*, 12(1):2 – 16, Feb. 2004.

[84] U. Ayesta, O. Brun, and B. J. Prabhu. Price of anarchy in non-cooperative load balancing games. *Performance Evaluation*, 68:1312–1332, 2011. Extended

version available as LAAS Research Report. Available at http://www.laas.fr/~brun/loadbalancing.pdf.

[85] Y. Korilis, A. Lazar, and A. Orda. Capacity allocation under noncooperative routing. *IEEE Transactions on Automatic Control*, 42(3):309–325, March 1997.

[86] A. Orda, R. Rom, and N. Shimkin. Competitive routing in multi-user communication networks. *IEEE/ACM Transactions on Networking*, 1:510–521, October 1993.

[87] M. Haviv and T. Roughgarden. The price of anarchy in an exponential multi-server. *Operations Research Letters*, 35:421–426, 2007.

[88] Jonatha Anselmi and Bruno Gaujal. Optimal routing in parallel, non-observable queues and the price of anarchy revisited. In *22nd International Teletraffic Congress (ITC)*, Amsterdam, 2010.

[89] H. L. Chen, J. Marden, and A. Wierman. The effect of local scheduling in load balancing designs. In *Proceedings of IEEE Infocom*, 2009.

[90] Olivier Brun and Balakrishna Prabhu. Worst-case Analysis of Non-Cooperative Load Balancing. Technical report, October 2012.

[91] R. Cominetti, J. R. Correa, and N. E. Stier-Moses. The impact of oligopolistic competition in networks. *Operations Research, Published online in Articles in Advance, DOI: 10.1287/opre.1080.0653*, June 2009.

[92] T. Roughgarden. The price of anarchy is independent of the network topology. *J. Comput. Syst. Sci.*, 67(2):341–364, 2003.

[93] Robert Margolies, Ashwin Sridharan, Vaneet Aggarwal, Rittwik Jana, N. K. Shankaranarayanan, Vinay A. Vaishampayan, and Gil Zussman. Exploiting mobility in proportional fair cellular scheduling: Measurements and algorithms. *IEEE/ACM Trans. Netw.*, 24(1):355–367, February 2016.

[94] Thi Thuy Nga Nguyen, Olivier Brun, and Balakrishna J. Prabhu. Using channel predictions for improved proportional-fair utility for vehicular users. *Comput. Networks*, 208:108872, 2022. https://hal.archives-ouvertes.fr/hal-02892099v2.

[95] Joan Boyar, Lene M. Favrholdt, Christian Kudahl, Kim S. Larsen, and Jesper W. Mikkelsen. Online algorithms with advice: A survey. *SIGACT News*, 47(3):93–129, aug 2016.

[96] Olivier Brun, Balakrishna J. Prabhu, and Tatiana Seregina. On the convergence of the best-response algorithm in routing games. In András Horváth, Peter Buchholz, Vittorio Cortellessa, Luca Muscariello, and Mark S. Squillante, editors, *7th International Conference on Performance Evaluation Methodologies and Tools, ValueTools '13, Torino, Italy, December 10-12, 2013*, pages 136–144. ICST/ACM, 2013.