Lyapunov-based design of a distributed Wardrop load balancing algorithm with application to Software Defined Networking

Antonio Pietrabissa*, Lorenzo Ricciardi Celsi*, Federico Cimorelli, Vincenzo Suraci,
Francesco Delli Priscoli, Alessandro Di Giorgio, Alessandro Giuseppi, and Salvatore Monaco

Abstract—**This paper presents an original discrete-time, distributed, non-cooperative load balancing algorithm, based on mean field game theory, which does not require explicit communications. The algorithm is proved to converge to an arbitrarily small neighborhood of a specific equilibrium among the loads of the providers, known as Wardrop equilibrium. Thanks to its characteristics, the algorithm is suitable for the Software Defined Networking (SDN) scenario, where service requests coming from the network nodes, i.e., the switches, are managed by the so-called SDN Controllers, playing the role of providers. The proposed approach is aimed at dynamically balancing the requests of the switches among the SDN Controllers to avoid congestion. The paper also suggests the adoption of SDN Proxies to improve the scalability of the overall SDN paradigm and presents an implementation of the algorithm in a proof-of-concept SDN scenario, which shows the effectiveness of the proposed solution with respect to the current approaches.**

***Index Terms*—Load balancing, Lyapunov design, Wardrop equilibrium, Software Defined Networks.**

NOMENCLATURE

$C = \{1,2,\dots,c\}$	Set of commodities
$l_p(x_p)$	Latency of provider p with load x_p
$\mathcal{L}(\boldsymbol{x})$	Candidate Lyapunov function
$r_{pq}[k]$	Migration rate from provider p to provider q at time k
V	Set of providers
$x_p^i, i \in C, p \in V$	Load of commodity i relying on provider p
$\boldsymbol{x} = (x_p^i), i \in C, p \in V$	Flow vector
$\mathcal{X}$	Feasible state space
$\mathcal{X}_{\mathcal{W}}$	Wardrop equilibrium set
$\mathcal{X}_{\mathcal{W},\varepsilon}$	ε-Wardrop equilibrium set
$\lambda^i, i = 1,\dots,c$	Per-commodity flow demand
$\mu_{pq}(l_p, l_q)$	Migration policy from provider p to provider q
$\Phi(\boldsymbol{x})$	Beckmann, McGuire and Winsten potential

I. INTRODUCTION

With the advent of cloud services, enterprises and carriers have found themselves in need of evolving their network infrastructures to satisfy new requirements, such as managing a higher demand for bandwidth and for responsiveness to new data patterns (including machine-to-machine, data-center and mobility traffic), scaling IT resources, sharing the same infrastructure among different logically isolated networks, and applying network-wide policies.

In order to face these new challenges, the current research is focused on the virtualization and elastic provisioning of the network resources within the Software Defined Networking (SDN) paradigm [1]. Such innovation is carried out by different public/private initiatives: among others, the FP7 T-NOVA and FI-Core research projects.

As defined by the Open Network Foundation (ONF), "SDN is a network architecture where network control is decoupled from forwarding and is directly programmable." While the so-called *forwarding plane* (in charge of physically routing the data packets) resides in the network nodes, i.e., in the switches, all the intelligence related to network control is logically centralized into a single software entity called SDN Controller, responsible for the control of the network behavior. In short, the SDN Controller is in charge of managing all network information, thus making, and enforcing into the underlying nodes, suitable forwarding decisions.

Since the network intelligence is shifted from the switches to the SDN Controller, each switch sends to the SDN Controller requests for forwarding decisions, each of which constitutes a *unit of workload*, or *job*. Although the SDN Controller can be vertically scaled, it can still saturate, thus becoming a relevant bottleneck for the network plant in terms of throughput and latency.

To overcome this scalability issue, a pool of SDN Controllers can be arranged in a cluster to form a physically distributed but logically centralized Control Plane sharing the overall network

This work is supported by the European Commission in the framework of the FP7 projects T-NOVA (*Network Functions as-a-Service over Virtualised Infrastructures*, www.t-nova.eu/) under Grant Agreement no. 619520, and FI-Core (*Future Internet - Core*) under Grant Agreement no. 632893.

A. Pietrabissa, L. Ricciardi Celsi, F. Cimorelli, F. Delli Priscoli, A. Di Giorgio, A. Giuseppi, and S. Monaco are with the Department of Computer, Control, and Management Engineering Antonio Ruberti, University of Rome La Sapienza, via Ariosto 25, 00162, Rome, Italy (email:{pietrabissa, ricciardicelsi, cimorelli, dellipriscoli, digiorgio, giuseppi, monaco}@diag.uniroma1.it).

V. Suraci is with the SMART Engineering Solutions & Technologies (SMARTEST) Research Center, eCampus University, Via Isimbardi 10, 22060, Novedrate (CO), Italy (e-mail: vincenzo.suraci@uniecampus.it).

* A. Pietrabissa and L. Ricciardi Celsi are joint first authors.

information, workload and control effort. Up to the authors' knowledge, a drawback of such an approach resides in the fact that, in the currently adopted proximity-based approaches, each switch (and, consequently, its workload) is statically associated with the closest SDN Controller. However, this static approach is not effective, especially when the workload is not evenly spatially distributed and/or is dynamic (which is always true in communication networks).

Therefore, load balancing mechanisms and algorithms are required to dynamically allocate the workload among all the available SDN Controller instances with the overall aim of optimizing the network performances. In [2], a switch migration protocol was devised for this purpose, yet such an approach requires an advanced load estimation mechanism for both SDN Controllers and switches, and an intra-controller protocol to guarantee liveness and safety features.

In this paper, the performance of the SDN Controllers is defined by a latency function, which describes how the response time of the SDN Controller grows with its workload. The objective of the load balancing algorithm is then to direct the requests of the switches to the SDN Controllers in such a way that the values of the latency functions of the SDN Controllers are equalized. The two main problems in the algorithm development are:

- the fact that the latency functions are not known (e.g., the load/delay curve of an SDN Controller depends on its specific hardware and software implementation);
- a distributed approach is needed since a centralized approach would require too much control traffic to exchange information among the SDN Controllers and potentially thousands of switches.

In this paper, a distributed, non-cooperative and dynamic load balancing algorithm is consequently developed on the ground of mean field game theory; specifically, the algorithm considers each request from a switch as an *agent* (whose decision is to determine the SDN Controller such a request has to be routed to), and is based on the measured response time of the SDN Controllers themselves: the algorithm is such that the agent decisions lead to an equilibrium, known in mean field game theory as Wardrop equilibrium, where the values of the latency functions of the SDN Controllers are equalized.

The main motivations behind this work are then (i) to prove, using Lyapunov arguments, how the difference equation governing the global state of the system (and macroscopically abstracting the microscopic evolution of the single agents involved) converges to an arbitrarily small neighborhood of a Wardrop equilibrium, and (ii) to show the effectiveness of such an approach through its application to a real SDN scenario.

The work presented in this paper was carried out within the FP7 T-NOVA project (www.t-nova.eu), aimed at extending the emerging concept of SDN to the efficient reconfiguration and elastic scaling of virtualized network functionalities (see Section II.B). Indeed, the proposed algorithm is embedded in the T-NOVA SDN Control Plane, allowing the management of virtual networks over data-center physical infrastructures. However, we further note that, since the algorithm is developed within the research framework of Wardrop load balancing and selfish routing (see Section II.A), it can be applied to several problems and scenarios other than the one considered here.

The paper is organized as follows: Section II presents the state of the art on Wardrop load balancing and on SDN networks, as well as the proposed novelties; Section III presents the algorithm and the convergence proof; Section IV shows some results of an SDN implementation; Section V draws the conclusions.

II. State of the Art and Proposed Innovations

A. Wardrop Load Balancing

Several load balancing approaches have been proposed in the literature: namely, in [3] it is suggested to classify load balancing algorithms as based on global, cooperative or non-cooperative approaches. Global algorithms require that each node, by means of an extensive interconnected system, transmits its current state to a centralized load balancer, which judiciously assigns a job to each resource, while simultaneously optimizing a specific objective (e.g., the response time of the entire system over all jobs). This is a classical approach that has been studied extensively using different techniques (e.g., nonlinear optimization) until it has been outperformed by the other two above-mentioned approaches. In cooperative algorithms, several decision makers agree upon making a coordinated decision so that each one of them operates at its own optimum. Instead, non-cooperative algorithms entail the presence of several decision makers which optimize their own response time independently of the others, since cooperation is not allowed. In such a case, a Nash equilibrium condition is reached where no decision maker can receive any further benefit by changing its own decision unilaterally. In other words, the stability of the network under said algorithms is analysed in terms of reaching a load distribution in which no single job can move to any other node with a lesser number of jobs.

Furthermore, load balancing algorithms can be classified as either static or dynamic. Static load balancing relies on the available knowledge of the application load, whereas dynamic load balancing algorithms are required for settings where the load distribution is not known *a priori* and succeed in performing their decision-making process based on the current state of the system, which is generally made available via feedback.

From the large body of literature on load balancing, we recall [4] and [5] as examples of centralized static cooperative load balancing, [6] and [7] as examples of centralized static non-cooperative load balancing, [8] and [9] as examples of centralized dynamic load balancing, and we also recall [10], which, instead, addresses the problem of distributed dynamic load balancing relying upon local cooperation among neighboring network nodes.

The scenario considered in this paper requires a non-cooperative dynamic load balancing approach. This kind of algorithms are widely investigated in game-theoretic frameworks, where the problem can be described as a dynamic load balancing game, in which users distribute their loads in a non-cooperative and selfish fashion [11] (in some applications, these algorithms are also referred to as selfish routing ones). Moreover, in this paper we consider a renowned game-theoretic traffic model due to Wardrop [12], introduced to represent road traffic with an infinite number of agents, each being responsible

for an infinitesimal amount of traffic. Within this framework, a certain amount of traffic, or *flow demand*, has to be routed from a given source to a given destination via a collection of paths. Each agent has the possibility to distribute its own flow among a set of admissible paths. The network is characterized by non-decreasing latency functions depending on the flows on the edges. A combination of flows such that the latencies of all the employed paths are minimal is called a Wardrop equilibrium for the network. Indeed, a Nash equilibrium is said to become a Wardrop equilibrium whenever the number of decision makers is assumed to be infinite [13].

Distributed algorithms designed to make concurrent users converge to some game-theoretical equilibrium conditions often rely on re-allocating resources in a round-based fashion (see, for instance, [11], [14], [15], and [16]). Some of these algorithms, in particular, are based on the concept of sampling the different strategies at each round, in order to guarantee a certain degree of exploration of the surrounding environment and, at the same time, to favor the use of the best strategies (exploitation). This last approach is used in several learning techniques (see, for instance the reinforcement learning approach in [17]), thus leading the algorithm to eventually converge to the optimal solution while providing acceptable solutions in the transitory phase. However, in communication networks it is preferred to distribute the flows more regularly by splitting the transmission flows associated with each user into smaller flows, each one directed to one of the available providers, in a rate-based load balancing fashion. On the other hand, such approaches are usually presented as continuous-time algorithms which cannot be seamlessly implemented in a real communication network and whose advantages are highlighted only from a methodological point of view (e.g., [14]).

In Section III, a discrete-time rate-based load balancing algorithm is designed, which retains the advantages of the rate-based approaches while being actually implementable. In particular, the algorithm is designed so as to dynamically learn a Wardrop equilibrium efficiently and in a distributed fashion; it can be regarded as a discrete-time version of the algorithm presented in [14] and, in Section III.B, is proven to converge to an equilibrium where all the latencies are equalized up to a given tolerance ε.

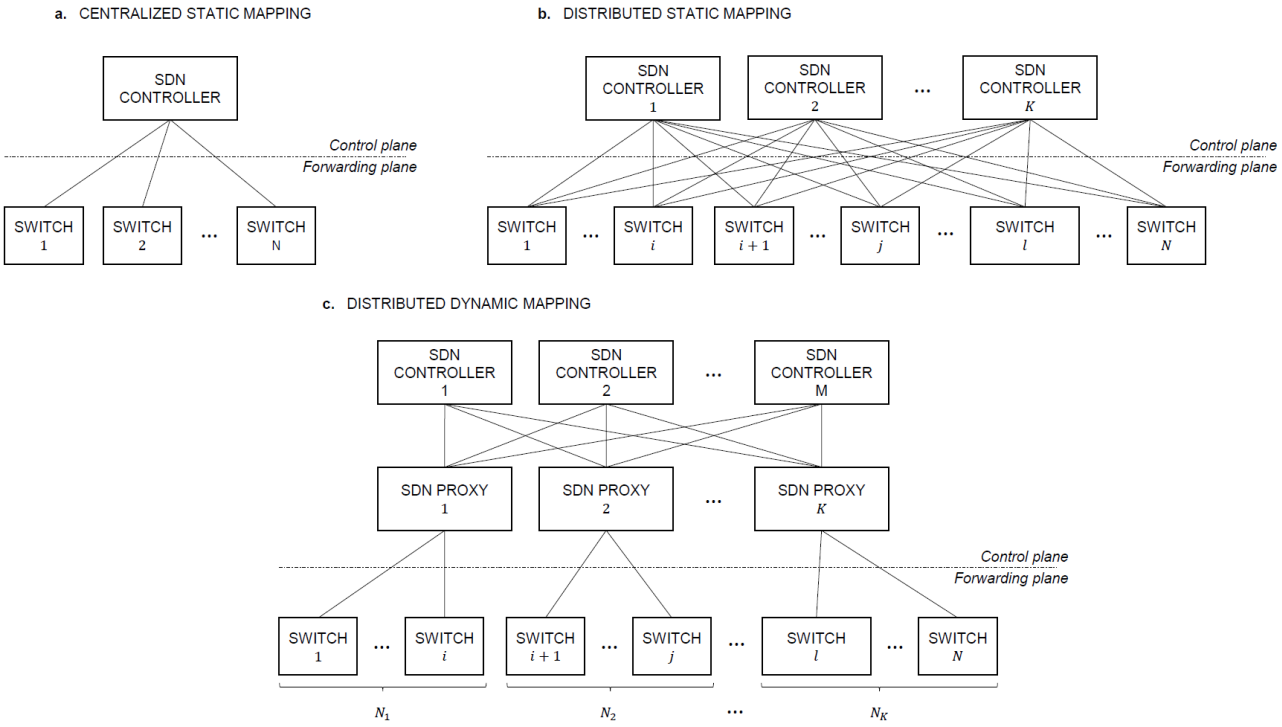

Fig. 1. SDN architectures: a. centralized static mapping, b. distributed static mapping and c. distributed dynamic mapping.

B. Software Defined Networks

The main SDN technology is defined by the OpenFlow standard [18], which is an open protocol of communication between the forwarding plane (i.e., the switches) and the control plane (i.e., the SDN Controller). The task of a switch is simply to interpret the forwarding rules sent by the SDN Controller, store them as forwarding tables and match incoming packets with the table entries.

The logical network architecture is shown in Fig. 1.a, where a single (logical) SDN Controller manages N switches. Since the logic is shifted from the switches to the SDN Controller, any forwarding decision consists of a unit of workload for the SDN Controller. For instance, according to [19], in large-scale scenarios (e.g., in a server cluster of $1.5 \cdot 10^3$ machines) the SDN workload has a mean arrival rate of about 10^5 OpenFlow messages per second. Therefore, an architectural improvement (Fig. 1.b) consists in physically distributing the SDN Control

Plane among K SDN Controllers, arranged to form a cluster, with a node playing the role of cluster coordinator [20].

A drawback of current Software Defined Networks resides in the fact that the mapping between the forwarding plane and the control plane, i.e., the association of each switch to a given SDN Controller, is static. In fact, in the literature, the problem of associating switches and SDN Controllers is known as "SDN Controller placement problem," and the proposed algorithms are mostly based on the location of the SDN Controllers with respect to the switches – see, e.g., [21] presenting a comparison between brute-force and greedy algorithm solutions, [22] presenting a heuristic approach, and [23] solving the problem by adopting operational research theory. Since, in real networks, the network workload varies in time, the main drawback of static mapping is the necessity to find a new solution to the controller placement problem whenever some SDN Controller workload exceeds its processing power (congestion occurrence).

To overcome this problem, a dynamic approach was preliminarily defined in [24], where each switch dynamically decides the SDN Controller to be associated with, based on the response time of the SDN Controllers. This suggested approach is however not compliant with the standard SDN protocols and is not scalable, since each switch is required to explicitly receive information about the current response time of all the SDN Controllers (instead, each switch is able to measure the response time only of the SDN Controller it is associated with).

Therefore, in this paper, a new entity is introduced, named SDN Proxy (see Fig. 1.c). The switches are statically connected to the nearest SDN Proxy. Each SDN Proxy receives the requests of its switches and has the task of forwarding them to one of the available SDN Controllers, based on a load balancing algorithm as the one proposed in this paper. The introduction of the SDN Proxies is transparent to the OpenFlow standard, as described in Section IV, and improves the SDN scalability since their tasks are quite simpler with respect to the ones of the SDN Controllers.

Hence, in order to manage the SDN Control Plane traffic in large-scale networks, by contrast with the currently adopted solutions relying on a logically centralized but physically distributed SDN Control Plane, the model proposed here considers the Control Plane as distributed across a cluster of multiple SDN Controller instances which are in charge of processing the network information and of enforcing suitable traffic control actions.

From a technological point of view, the OpenDaylight SDN Controller [25], an open-source project sponsored by the Linux Foundation and a large consortium of networking companies, has introduced a cluster-based implementation: it runs on a cluster of machines sharing a distributed data store to maintain the global view of the network. At the same time, the OpenFlow protocol, since version 1.3, has dictated the corresponding architectural model by introducing the concept of *Controller* for a switch, namely with a *Master*, *Slave* or *Equal* role. This enables two modes of operation when multiple SDN Controllers exist in a network: *Master/Slave Interaction* and *Equal Interaction* between the switch nodes and the cluster of SDN Controllers. In Master/Slave Interaction, each switch can be associated with multiple SDN Controllers but is managed by only one (the Master), being responsible for all the control actions for that switch, whereas the others (the Slaves) are used as backup controllers. In Equal Interaction, each switch can be associated with more than one SDN Controller instance, and have more than one Master association, that is, several Equal associations managing the switch.

III. Proposed Wardrop Load Balancing Algorithm

Section III.A describes the basic definitions needed for the algorithm analysis; Section III.B presents the load balancing algorithm and the convergence proof; Section III.C considers some implementation issues.

A. Preliminaries on Wardrop Equilibrium and on Set Stability

As anticipated in Section II, this paper further develops a well-known model for selfish routing [14], where an infinite population of agents carries an infinitesimal amount of load each, following the previous works [24] and [26] concerning distributed load balancing algorithms.

Let $C = \{1,2, \dots c\}$ denote a set of commodities with flow demands, or rates, $\lambda^i > 0, i \in C$, generally expressed in jobs per unit of time. For the sake of simplicity, each commodity $i \in C$ is associated with a (source, destination) couple of nodes, denoted with (s^i, d^i). The λ^i's are also such that $\sum_{i=1}^{c} \lambda^i = \lambda$. Let V denote a set of providers, which are used to transmit the flows for every commodity $i \in C$. All source nodes are connected by the network to the available providers, which, in turn, connect them to the destination nodes. As an example, we may think of the considered model as a description of a network consisting of a set of edges, over which the controllers arrange proper paths to connect source and destination nodes. In the considered scenario, the SDN Controllers are the providers, the SDN Proxies identify the commodities, and the λ^i's are their request loads, expressed in requests per unit of time.

The definition of *agent* is also required. As defined, for instance, in [15], each agent is an infinitesimal portion of a specified commodity, whose objective is to minimize the cost sustained to reach its destination by a proper flow assignment. In the considered scenario, a single request of the flow is approximately considered as an agent: in fact, even if the number of requests is finite, if the flow rates are sufficiently high, the population acceptably approximates the infinite population constraint required by Wardrop theory (see [14]).

Let x_p^i be the volume of the agents, or bandwidth, of commodity i relying on a provider $p \in V$. The vector $\boldsymbol{x} = \left(x_p^i\right)_{p \in V, i \in C}$ is the *flow vector* (in the literature also referred to as *population share* or *job vector)*, describing the overall amount of jobs per unit or time of commodity i.

Definition 1 (Feasible states). The feasible state space, i.e., the closed set of feasible job vectors, is

$$\mathcal{X} \coloneqq \left\{\boldsymbol{x} \in \mathbb{R}^{|V|\times|C|} \mid x_p^i \geq 0, \forall p \in V, \sum_{p \in V} x_p^i = \lambda^i, \forall i \in C\right\}, \quad (1)$$

where the λ^i's are the transmission rates required by each commodity i. ■

Let $x_p \coloneqq \sum_{i \in C} x_p^i$ denote the load of provider $p \in V$, and let

each provider be characterized by a continuous cost function, referred to as *latency function* and denoted with $l_p(\cdot)\colon [0,\lambda] \to \mathbb{R}_+$. The latency of a provider p is a function of its load x_p, i.e., $l_p(x_p)$ is the latency of controller p with load x_p.

An instance of the load balancing game is then

$$\Gamma = \left\{V, (l_p)_{p\in V}, (s^i, d^i, \lambda^i)_{i\in C}\right\}. \quad (2)$$

The load balancing problem is formulated below as the problem of determining the strategies which will lead the flow vector to reach an arbitrarily small neighborhood of a Wardrop equilibrium. In Wardrop theory, *stable* flow assignments are the ones in which no agent (i.e., no "small" portion of a commodity directed from a source to a destination) can improve its situation by changing its strategy (i.e., the set of used providers) unilaterally. This objective is achieved if all agents reach a Wardrop equilibrium.

Definition 2 (*Wardrop equilibrium*, [14]). A feasible flow vector $\boldsymbol{x}$ is at a Wardrop equilibrium for the instance Γ of the load balancing game if, for each provider $p \in V$ such that $x_p > 0$, the following relation holds: $l_p(x_p) \le l_q(x_q), \forall q \in V$. The set of all Wardrop equilibria is the following subset of $\mathcal{X}$:

$$\mathcal{X}_{\mathcal{W}} \coloneqq \{\boldsymbol{x} \in \mathcal{X} \mid l_p(x_p) - l_q(x_q) \le 0,\ x_p > 0, \forall p, q \in V\}. \quad (3)$$

■

In practice, at the Wardrop equilibrium, the latencies of all the loaded providers have the same value: therefore, provided that the latency functions properly represent the provider performances, a fair exploitation of the resources is achieved by driving the flows towards a Wardrop equilibrium.

In the framework of researches on Wardrop equilibria , a key role is played by the Beckmann, McGuire, and Winsten potential [27], given by:

$$\Phi(\boldsymbol{x}) \coloneqq \sum_{p\in V} \int_0^{x_p} l_p(s)ds, \quad (4)$$

whose properties are summarised in Property 1 below, under mild assumptions on the l_p's.

Assumption 1 (Latency functions). The latency function l_p exhibits the following properties, $\forall p \in V$:

a) $l_p(x)$ is positive and non-decreasing for $x \in [0,\lambda]$;
b) $l_p(x)$ is Lipschitz continuous for $x \in [0,\lambda]$. ■

Property 1 ([28], [29]). Under Assumption 1, the potential (4) is continuous and has the following properties:

a) a flow minimizes Φ if and only if no agent can improve its own latency, implying that the set of Wardrop equilibria coincides with the set of flows minimizing Φ;
b) at least one positive minimum Φ_{min} of Φ over the set of feasible flows (and thus at least one Wardrop equilibrium) exists;
c) if the latency functions are strictly increasing, the minimizing flow is unique;
d) if the latency functions are strictly increasing and $l_p(0) = l_q(0), \forall p, q \in V$, the unique minimum of Φ is achieved when the latencies of all the providers are equalized. ■

The following definition and theorem on set stability (i.e., on the stability of a set of points in the state space) are also recalled with respect to the nonlinear discrete-time dynamics

$$\boldsymbol{x}[k+1] = f(\boldsymbol{x}[k]), \quad \boldsymbol{x}(0) \in \mathcal{X}. \quad (5)$$

Definition 3 (*Positive definiteness*, [30]). Let $\mathcal{X}$ be an invariant set for system (5), let $\mathcal{A}$ be a closed subset of $\mathcal{X}$ and let $d(\boldsymbol{x}, \mathcal{A}) \coloneqq \inf_{\boldsymbol{y}\in\mathcal{A}} |\boldsymbol{x} - \boldsymbol{y}|$ be the distance from a point $\boldsymbol{x} \in \mathcal{X} \setminus \mathcal{A}$ to $\mathcal{A}$. The function $\mathcal{L}(\boldsymbol{x})\colon \mathcal{X} \to \mathbb{R}_+$ is positive definite with respect to the set $\mathcal{A} \subset \mathcal{X}$ if there exists an increasing continuous function $\psi\colon \mathbb{R}_+ \to \mathbb{R}_+$ such that $\psi(0) = \psi_{min}$ and $\psi(d(\boldsymbol{x}, \mathcal{A})) \le \mathcal{L}(\boldsymbol{x}), \forall \boldsymbol{x} \in \mathcal{X} \setminus \mathcal{A}$. ■

Let $\Delta\mathcal{L}(\boldsymbol{x}[k]) \coloneqq \mathcal{L}(\boldsymbol{x}[k+1]) - \mathcal{L}(\boldsymbol{x}[k])$ denote the difference of a Lyapunov function $\mathcal{L}(\boldsymbol{x})$ along the solutions of system (5). Lyapunov's second method can be applied to verify if a set is a Globally Asymptotically Stable Set (GASS) as follows [30].

Theorem 1 (Globally Asymptotically Stable Set). Given a closed subset $\mathcal{A} \subset \mathcal{X}$ and a Lyapunov function $\mathcal{L}(\boldsymbol{x})$ in $\mathcal{X} \setminus \mathcal{A}$, if $\mathcal{L}(\boldsymbol{x})$ and $-\Delta\mathcal{L}(\boldsymbol{x}[k])$ are positive definite with respect to $\mathcal{A}$, then $\mathcal{A}$ is a GASS for system (5). ■

B. Load Balancing Algorithm and Convergence Proof

1) Load Balancing Algorithm

Let the system dynamics (5) be expressed component-wise by

$$x_p^i[k+1] = x_p^i[k] + \delta\left(\sum_{q\in V} r_{qp}^i[k] - \sum_{q\in V} r_{pq}^i[k]\right), \quad \forall p \in V, \forall i \in C, k = 0,1,\ldots, \quad (6)$$

where δ is the sampling period and $r_{pq}^i[k]$ is the so-called *migration rate* from provider p to provider q. Inspired by the continuous-time algorithm in [14], the migration rate is defined as:

$$r_{pq}^i[k] = x_p^i[k]\sigma_{pq}^i[k]\mu_{pq}^i\left(l_p(x_p[k]), l_q(x_q([k])\right), \quad \forall p, q \in V, \forall i \in C, k = 0,1,\ldots, \quad (7)$$

where $\sigma_{pq}^i[k]$ is the control gain, which sets the rate with which the population share of provider p migrates to provider q, and $\mu_{pq}(l_p, l_q)$ is the so-called *migration policy*, representing the decision whether (and in which percentage) the population share assigned to provider p migrates to provider q.
The proposed migration policy has the following property:

$$\begin{cases} \mu_{pq}^i(l_p, l_q) = 0, & \text{if } l_p \le l_q + \varepsilon, \\ \mu_{pq}^i(l_p, l_q) \in \left[\underline{\mu}, \bar{\mu}\right], 0 < \underline{\mu} < \bar{\mu} < +\infty, & \text{otherwise,} \end{cases}$$

$$\forall p, q \in V, \forall i \in C, \varepsilon > 0, \quad (8)$$

where ε is a tolerance on the maximum acceptable mismatch between the couples of latency values and $\underline{\mu}$ and $\bar{\mu}$ are positive lower- and upper-bounds, respectively. As shown in the following, the tolerance ε is introduced since the usual migration policies adopted in the continuous-time algorithms (obtained from (8) by setting $\varepsilon = 0$) cannot guarantee convergence in the discrete-time case, however small the sampling period (see, e.g., [29]).

Let the total migration rate from provider p to provider q be defined as $r_{pq}[k] \coloneqq \sum_{i \in C} r_{pq}^i[k]$. For notational simplicity, whenever unambiguous, $\mu_{pq}^i[k]$ will be used in place of $\mu_{pq}^i\big(l_p(x_p[k]), l_q(x_q([k]))\big)$.

2) Convergence Proof

Before analysing the algorithm convergence, the following definition of ε-Wardrop equilibrium is introduced.

Definition 4 (ε-Wardrop equilibrium). A feasible flow vector $\boldsymbol{x} = \left(x_p^i\right)_{p \in V, i \in C}$ is defined to be at an ε-Wardrop equilibrium for the instance Γ of the load balancing game if, for each provider $p \in V$ such that $x_p > 0$, the following relation holds: $l_p(x_p) \leq l_u(x_u) + \varepsilon, \forall u \in V$, for $0 < \varepsilon < \bar{l} - \underline{l}$, where $\bar{l} \coloneqq \max_{p \in V} l_p(\lambda)$ and $\underline{l} \coloneqq \min_{p \in V} l_p(0)$ are the maximum and minimum latency values, respectively. The set of all ε-Wardrop equilibria is the following closed subset of $\mathcal{X}$:

$$\mathcal{X}_{\mathcal{W},\varepsilon} \coloneqq \left\{ \boldsymbol{x} \in \mathcal{X}, \varepsilon > 0 \;\middle|\; l_p(x_p) \leq l_j(x_j) + \varepsilon,\ x_p > 0, \forall j \in V, \forall p \in V,\ 0 < \varepsilon < \bar{l} - \underline{l} \right\}. \quad (9)$$

■

In practice, at an ε-Wardrop equilibrium, the latencies of all the loaded providers have the same value up to the tolerance ε.

Hereafter, the following assumptions on the latency functions and on the migration policy (8) will be considered.

Assumption 2. The latency functions, the migration policy (8) and the control gain exhibit the following properties:

a) $l_p(x)$ is increasing with x, for $x \in [0, \lambda]$, $\forall p \in V$;
b) $l_p(x)$ is locally Lipschitz continuous, $\forall p \in V$, $\forall x \in [0, \lambda]$; with Lipschitz constant $\eta_p(x)$; let the maximum Lipschitz constant be defined as $\bar{\eta} \coloneqq \max_{p \in V,\ x \in [0,\lambda]} \eta_p(x)$;
c) $\mu_{pq}^i\left(l_p, l_q\right)$ is Lipschitz-continuous, $\forall l_p, l_q \in [\underline{l}, \bar{l}]$, $\forall i \in C$;
d) $\sigma_{pq}^i[k]$ is constant and equal to $\sigma = \frac{\varepsilon}{|V| \bar{\lambda} \bar{\eta} \bar{\mu} \delta}$, where $\bar{\lambda}$ is a flow rate upper bound;
e) $\varepsilon < \bar{l} - \underline{l}$. ■

Assumptions 2.a) and 2.b) are slightly more restrictive than Assumption 1. Assumption 2.a), introduced for the sake of simplicity in the system analysis, yields that, by Property 1, the Wardrop equilibrium and the corresponding flow vector, denoted with $l_{\mathcal{W}}$ and $\boldsymbol{x}_{\mathcal{W}}$, respectively, are unique. Assumption 2.b) states that limited population differences lead to limited differences in the latency values.

Definition 5 (distance). Let the norm of a state be defined as $\|\boldsymbol{x}\| \coloneqq \max_{p \in V | x_p > 0} \left(l_p(x_p) - l_{\mathcal{W}}\right)$, and let the distance between a state $\boldsymbol{x} \in \mathcal{X} \setminus \mathcal{X}_{\mathcal{W},\varepsilon}$ and the set $\mathcal{X}_{\mathcal{W},\varepsilon}$ be defined as $d\left(\boldsymbol{x}, \mathcal{X}_{\mathcal{W},\varepsilon}\right) \coloneqq \|\boldsymbol{x}\| - \max_{\boldsymbol{y} \in \mathcal{X}_{\mathcal{W},\varepsilon}} \|\boldsymbol{y}\| > 0$. ■

Under Assumption 2, the set of Wardrop and ε-Wardrop equilibria can be written as:

$$\mathcal{X}_{\mathcal{W}} \coloneqq \{\boldsymbol{x} \in \mathcal{X} \mid \|\boldsymbol{x}\| = 0\} = \{\boldsymbol{x}_{\mathcal{W}}\}; \quad (10)$$

$$\mathcal{X}_{\mathcal{W},\varepsilon} \coloneqq \{\boldsymbol{x} \in \mathcal{X}, \varepsilon > 0 \mid \|\boldsymbol{x}\| \leq \varepsilon\}. \quad (11)$$

and the following properties hold.

Property 2. The feasible set $\mathcal{X}$, and the set of the ε-Wardrop equilibria $\mathcal{X}_{\mathcal{W},\varepsilon}$ are such that:

(P1) $\mathcal{X}_{\mathcal{W},\varepsilon} = \mathcal{X}$ if $\varepsilon \geq \bar{l} - \underline{l}$;
(P2) $\mathcal{X} \supset \mathcal{X}_{\mathcal{W},\varepsilon_2} \supset \mathcal{X}_{\mathcal{W},\varepsilon_1} \supset \{\boldsymbol{x}_{\mathcal{W}}\}, \forall \varepsilon_1, \varepsilon_2 \mid 0 < \varepsilon_1 < \varepsilon_2 < \bar{l} - \underline{l}$;
(P3) $\mathcal{X}_{\mathcal{W},\varepsilon} \to \{\boldsymbol{x}_{\mathcal{W}}\}$ as $\varepsilon \to 0$. ■

The set convergence to an arbitrarily small neighborhood of the Wardrop equilibrium is proven by using the Beckmann, McGuire, and Winsten potential (4) to build a candidate Lyapunov function. The following lemma demonstrates some properties of the potential which will be used in the convergence proof of the subsequent Theorem 2.

Lemma 1 (Properties of the potential). Under Assumption 2, the following properties hold for the nonlinear discrete-time system (3), (6), (7), (8), with total flow rate $\lambda > 0$:

(L1) $\Phi(\boldsymbol{x}) > \Phi_{min}, \forall \boldsymbol{x} \in \mathcal{X} \setminus \{\boldsymbol{x}_{\mathcal{W}}\}$, $\Phi(\boldsymbol{x}_{\mathcal{W}}) = \Phi_{min}$;
(L2) $\Phi(\boldsymbol{x}[k+1]) - \Phi(\boldsymbol{x}[k]) < 0, \forall \boldsymbol{x}[k] \in \mathcal{X} \setminus \mathcal{X}_{\mathcal{W},\varepsilon}$;
(L3) $\Phi(\boldsymbol{x}[k+1]) - \Phi(\boldsymbol{x}[k]) = 0, \forall \boldsymbol{x}[k] \in \mathcal{X}_{\mathcal{W},\varepsilon}$. ■

Proof. Conditions (L1) hold thanks to Property 1.

To verify condition (L2), the variation of the Lyapunov function along any trajectory of the considered system is written as follows:

$$\begin{aligned}
&\Delta\Phi(\boldsymbol{x}[k]) = \Phi(\boldsymbol{x}[k+1]) - \Phi(\boldsymbol{x}[k]) = \\
&\sum_{p \in V} \int_{x_p[k]}^{x_p[k+1]} l_p(s) ds \\
&\leq \sum_{p \in V} \left(x_p[k+1] - x_p[k]\right) l_p\left(x_p[k+1]\right) \\
&= \sum_{p \in V} \left(\delta \sum_q r_{qp}[k] - \delta \sum_q r_{pq}[k]\right) l_p\left(x_p[k+1]\right) \\
&= \delta\Big(\sum_{p \in V} \sum_{q \in V} r_{qp}[k]\, l_p\left(x_p[k+1]\right) - \\
&\sum_{p \in V} \sum_{q \in V} r_{pq}[k]\, l_p\left(x_p[k+1]\right)\Big) \\
&= \delta\Big(\sum_{p \in V} \sum_{q \in V} r_{pq}[k]\, l_q\left(x_q[k+1]\right) - \\
&\sum_{p \in V} \sum_{q \in V} r_{pq}[k]\, l_p\left(x_p[k+1]\right)\Big) \\
&= \delta\left(\sum_{p \in V} \sum_{q \in V} r_{pq}[k] \left(l_q\left(x_q[k+1]\right) - l_p\left(x_p[k+1]\right)\right)\right), \quad (12)
\end{aligned}$$

where the inequality holds from geometrical considerations (see Appendix A).

The following shows that (i), if $l_p(x_p[k]) - l_q(x_q[k]) > \varepsilon$, with $x_p[k] > 0$, the corresponding term of the summation in the last row of (12) is negative, whereas (ii), if $l_p(x_p[k]) - l_q(x_q[k]) \leq \varepsilon$ or $x_p[k] = 0$, the term is null.

(i) If $l_p(x_p[k]) - l_q(x_q[k]) > \varepsilon$, with $x_p[k] > 0$, from Assumptions 2.a)-2.b) it follows that $l_p(x_p[k]) > 0$. Moreover, $\mu_{pq}^i[k] > 0$ from equation (8) and, thus, $r_{pq}^i[k] > 0, \forall i \in C$. Now we need to show that $l_p(x_p[k+1]) - l_q(x_q[k+1]) > 0$. From equation (6) the following inequality holds:

$$l_p(x_p[k+1]) - l_q(x_q[k+1]) \geq l_p(x_p[k] - \delta \sum_{q\in V} r_{pq}[k]) - l_q(x_q[k] + \delta \sum_{p\in V} r_{pq}[k]), \forall p, q \in V, k = 0,1, ... \quad (13)$$

In equation (13), the worst-case system dynamics over δ is considered, in which no commodities migrate part of their population to provider p and from provider q. From Assumption 2.b), it holds that

$$\begin{cases} l_p(x_p[k] - \delta \sum_{q\in V} r_{pq}[k]) \geq l_p(x_p[k]) - \bar{\eta}\delta \sum_{q\in V} r_{pq}[k] \\ l_q(x_q[k] + \delta \sum_{p\in V} r_{pq}[k]) \leq l_q(x_q[k]) + \bar{\eta}\delta \sum_{p\in V} r_{pq}[k]. \end{cases} \quad (14)$$

Since we are analysing the case $l_p(x_p[k]) - l_q(x_q[k]) > \varepsilon$, the following inequality holds:

$$l_p(x_p[k+1]) - l_q(x_q[k+1]) \geq \varepsilon - \bar{\eta}\delta(\sum_{q\in V} r_{pq}[k] + \sum_{p\in V} r_{pq}[k]). \quad (15)$$

From equation (7) and Assumption 2.a), and considering that $x_p^i[k] \leq \lambda^i, \forall i \in C$, the following upper-bound holds:

$$r_{pq}^i[k] = x_p^i[k]\sigma_{pq}^i[k]\mu_{pq}^i[k] \leq \lambda^i \sigma\bar{\mu}, \forall p, q \in V, \forall i \in C. \quad (16)$$

Then, considering that there are at most $(|V| - 1)$ terms in the first summation of the second term of equation (15), the summation is upper-bounded by

$$\sum_{q\in V} r_{pq}[k] = \sum_{i\in C} \sum_{q\in V} r_{pq}^i[k] \leq \sigma\bar{\mu}(|V| - 1)\lambda. \quad (17)$$

The second summation of the second term of (15) is upper-bounded by:

$$\sum_{p\in V} r_{pq}[k] = \sum_{i\in C} \sum_{p\in V} r_{pq}^i[k] \leq \sigma\bar{\mu} \sum_{i\in C} \sum_{p\in V} x_p^i[k] = \sigma\bar{\mu}\lambda. \quad (18)$$

From equations (17) and (18), we obtain that a sufficient condition for the right-hand side of equation (15) to be non-negative is the following:

$$\varepsilon \geq |V|\sigma\lambda\bar{\mu}\bar{\eta}\delta, \quad (19)$$

which holds by Assumption 2.e).

(ii) If $l_p(x_p[k]) - l_q(x_q[k]) \leq \varepsilon$ or $x_p[k] = 0$, $r_{pq}[k] = 0$ by equations (7) and (8), and the corresponding term of the summation in the last row of (12) is null.

From (i) and (ii) it follows that property (L2) holds since, if $\boldsymbol{x}[k] \notin \mathcal{X}_{\mathcal{W},\varepsilon}$ (i.e., there exists at least one couple $(p, q) \in V^2$ such that $l_p(x_p[k]) - l_q(x_q[k]) > \varepsilon$, with $x_p[k] > 0$), at least one term of equation (12) is negative; property (L3) holds since, if $\boldsymbol{x}[k] \in \mathcal{X}_{\mathcal{W},\varepsilon}$ (i.e., for all couples $(p, q) \in V^2$ with $x_p[k] > 0$ we have that $l_p(x_p[k]) - l_q(x_q[k]) \leq \varepsilon$), all the terms of equation (12) are null. ■

Theorem 2 (ε-Wardrop equilibrium set as a GASS). Under Assumption 2, $\mathcal{X}_{\mathcal{W},\varepsilon}$ is a GASS for the nonlinear discrete-time system (3), (6), (7), (8), with total flow rate $\lambda > 0$. ■

Proof. The proof is structured as follows: first, it is shown that the feasible state space is a positively invariant set (A); secondly, the asymptotic set stability is proven (B).

(A) *Feasibility*.

It is shown in the following that, since the initial job vector is feasible (i.e., from Definition 1, $\sum_{p\in V} x_p^i[0] = \lambda^i$ and $x_p^i[0] \geq 0, \forall p \in V, \forall i \in C$), the job vector is feasible during the entire system dynamics. In fact, it follows from equation (6) that:

$$\sum_{p\in V}(x_p^i[k+1] - x_p^i[k]) = \sum_{p\in V} \sum_{q\in V}(r_{qp}^i[k] - r_{pq}^i[k])\delta = \sum_{p\in V} \sum_{q\in V} r_{qp}^i[k] \cdot \delta - \sum_{q\in V} \sum_{p\in V} r_{qp}^i[k]\delta = 0, \quad (20)$$

and, therefore, that $\sum_{p\in V} x_p^i[k] = \sum_{p\in V} x_p^i[0] = \lambda^i, \forall k \geq 0$.

By induction, since $x_p^i[0] \geq 0, \forall p \in V$, and given equation (6), in order to prove that $x_p^i[k] \geq 0, \forall k \geq 0$, it is sufficient to assume that $x_p^i[k] \geq 0, \forall p \in V, \forall i \in C$, for a given k, and to prove that

$$x_p^i[k+1] = x_p^i[k] + \delta \sum_{q\in V}(r_{qp}^i[k] - r_{pq}^i[k]) \geq 0, \forall p \in V, \forall i \in C. \quad (21)$$

In this respect, it can be observed that the following inequality holds (considering that, in the worst-case, no commodities migrate part of their population to provider p):

$$x_p^i[k+1] \geq x_p^i[k] - \delta \sum_{q\in V} r_{pq}^i[k], \forall p \in V, \forall i \in C. \quad (22)$$

From definition (8) it follows that $r_{pp}^i[k] = 0$, so there are at most $(|V| - 1)$ terms in the summation of equation (22). Thus, considering that $r_{pq}^i[k] \leq x_p^i[k]\sigma\bar{\mu}$, the condition in (22) is met if the following inequality holds:

$$x_p^i[k] - x_p^i[k](|V| - 1)\sigma\bar{\mu}\delta \geq 0, \forall p \in V, \forall i \in C. \quad (23)$$

If $x_p[k] = 0$, inequality (23) is verified. If $x_p[k] > 0$, inequality (23) is verified provided that:

$$\sigma \leq \frac{1}{(|V|-1)\delta\bar{\mu}}, \quad (24)$$

which holds by Assumption 2.e), considering that $\frac{\varepsilon}{\bar{\eta}\bar{\lambda}} < 1$ (in fact, by the definitions of $\bar{\eta}$ and $\bar{l}$, it holds that $\bar{\eta}\bar{\lambda} \geq \bar{l}$, and, by Assumption 2.f), it holds that $\bar{l} > \varepsilon$).

(B) *Global asymptotic set stability*.

Let $\mathcal{L}(\boldsymbol{x}) \coloneqq \Phi(\boldsymbol{x}) - \Phi_{min}$ be the candidate Lyapunov function, where $\Phi(\boldsymbol{x})$ is the potential (4) and Φ_{min} is its minimum value, which is unique thanks to Assumption 2.

If $\boldsymbol{x} \in \mathcal{X}_{\mathcal{W},\varepsilon}$, from Lemma 1 it follows that $\mathcal{L}(\boldsymbol{x})$ is positive definite and that $\Delta\mathcal{L}(\boldsymbol{x}[k]) = 0$. If $\boldsymbol{x} \in \mathcal{X} \setminus \mathcal{X}_{\mathcal{W},\varepsilon}$, it is shown below that $\mathcal{L}(\boldsymbol{x})$ and $-\Delta\mathcal{L}(\boldsymbol{x}[k])$ are positive definite with respect to the closed set $\mathcal{X}_{\mathcal{W},\varepsilon}$.

B1. $\mathcal{L}(\boldsymbol{x})$ is positive definite with respect to $\mathcal{X}_{\mathcal{W},\varepsilon}$

Let $\psi: \mathbb{R}_+ \to \mathbb{R}_+$ be defined as follows: $\psi\big(d(\boldsymbol{x}, \mathcal{X}_{\mathcal{W},\varepsilon})\big) \coloneqq \gamma_1 d\big(\boldsymbol{x}, \mathcal{X}_{\mathcal{W},\varepsilon}\big)$, with $\gamma_1 > 0$. By definition, we have that $\psi(0) = 0$ and that $\psi\big(d(\boldsymbol{x}, \mathcal{X}_{\mathcal{W},\varepsilon})\big)$ is increasing with $d(\boldsymbol{x}, \mathcal{X}_{\mathcal{W},\varepsilon})$. We have to show that $\psi\big(d(\boldsymbol{x}, \mathcal{X}_{\mathcal{W},\varepsilon})\big) = \gamma_1 d\big(\boldsymbol{x}, \mathcal{X}_{\mathcal{W},\varepsilon}\big) \leq \mathcal{L}(\boldsymbol{x}) = \Phi(\boldsymbol{x}) - \Phi_{min}, \forall \boldsymbol{x} \in \mathcal{X} \setminus \mathcal{X}_{\mathcal{W},\varepsilon}$, i.e., that a value for γ_1 exists such that the following inequality holds:

$$\gamma_1 \leq \frac{\Phi(\boldsymbol{x}[k]) - \Phi_{min}}{d(\boldsymbol{x}[k], \mathcal{X}_{\mathcal{W},\varepsilon})}, \forall k = 0,1,2, \ldots$$

Since $\boldsymbol{x} \in \mathcal{X} \setminus \mathcal{X}_{\mathcal{W},\varepsilon}$, by definition it holds that $d\big(\boldsymbol{x}, \mathcal{X}_{\mathcal{W},\varepsilon}\big) > 0$. By Assumption 2.d), $d\big(\boldsymbol{x}, \mathcal{X}_{\mathcal{W},\varepsilon}\big)$ is upper-bounded by $(\bar{l} - \underline{l})$. From geometrical considerations (see Appendix B), it turns out that $\Phi(\boldsymbol{x}) - \Phi_{min} > \frac{\varepsilon^2}{4\bar{\eta}} > 0, \forall \boldsymbol{x} \in \mathcal{X} \setminus \mathcal{X}_{\mathcal{W},\varepsilon}$. Therefore, a suitable choice for γ_1 is $\gamma_1 = \frac{\varepsilon^2}{4\bar{\eta}(\bar{l}-\underline{l})}$.

B2. $-\Delta\mathcal{L}(\boldsymbol{x}[k])$ is positive definite with respect to $\mathcal{X}_{\mathcal{W},\varepsilon}$

Let $\psi: \mathbb{R}_+ \to \mathbb{R}_+$ be defined as $\psi\big(d(\boldsymbol{x}, \mathcal{X}_{\mathcal{W},\varepsilon})\big) \coloneqq \gamma_2\, d\big(\boldsymbol{x}, \mathcal{X}_{\mathcal{W},\varepsilon}\big)$, with $\gamma_2 > 0$. Also, consider that $\Delta\mathcal{L}(\boldsymbol{x}[k]) = \Delta\Phi(\boldsymbol{x}[k])$ and that, from Lemma 1, $\Delta\Phi(\boldsymbol{x}[k]) < 0, \forall \boldsymbol{x}[k] \in \mathcal{X} \setminus \boldsymbol{x}_{\mathcal{W}}$.

We have to show that $\psi\big(d(\boldsymbol{x}[k], \mathcal{X}_{\mathcal{W},\varepsilon})\big) = \gamma_2\, d\big(\boldsymbol{x}, \mathcal{X}_{\mathcal{W},\varepsilon}\big) \leq -\Delta\mathcal{L}(\boldsymbol{x}[k]) = -\Delta\Phi(\boldsymbol{x}[k]), \forall \boldsymbol{x}[k] \in \mathcal{X} \setminus \mathcal{X}_{\mathcal{W},\varepsilon}$, i.e., that there exists a value for γ_2 such that the following inequality holds:

$$\gamma_2 \leq \frac{-\Delta\Phi(\boldsymbol{x}[k])}{d(\boldsymbol{x}[k], \mathcal{X}_{\mathcal{W},\varepsilon})}, \forall k = 0,1,2, \ldots \quad (25)$$

For the numerator of equation (25), the following inequality holds (see Lemma 1, equation (12)):

$$-\Delta\Phi(\boldsymbol{x}[k]) = -\sum_{p\in V} \int_{x_p[k]}^{x_p[k+1]} l_p(s)ds \geq$$
$$\delta \sum_{p\in V} \sum_{q\in V} r_{pq}[k] \Big(l_p\big(x_p[k+1]\big) - l_q\big(x_q[k+1]\big)\Big), \forall k = 1,2, \ldots \quad (26)$$

where the terms of the last summation are either null or positive. From equations (15) and (19) it follows that $l_p\big(x_p[k+1]\big) - l_q\big(x_q[k+1]\big) \geq 0$. Let us consider the loaded provider p^* which has the maximum latency value at time k, i.e., $p^* = \text{argmax}_{p\in V |\, x_p[k]>0} l_p\big(x_p[k]\big)$. We thus write from equations (26), (7) and (8):

$$-\Delta\Phi(\boldsymbol{x}[k]) \geq \delta x_{p^*}[k]\sigma\underline{\mu}. \quad (27)$$

Since $\boldsymbol{x}[k] \in \mathcal{X} \setminus \mathcal{X}_{\mathcal{W},\varepsilon}$, we know that $l_{p^*}\big(x_{p^*}[k]\big) - l_q\big(x_q[k]\big) > \varepsilon$, for at least one $q \in V$, yielding $l_{p^*}\big(x_{p^*}[k]\big) > \varepsilon$. Assumption 2.b) yields that $x_{p^*}[k] \geq \frac{l_{p^*}\left(x_{p^*}[k]\right)}{\bar{\eta}} > \frac{\varepsilon}{\bar{\eta}}$. Thus, recalling that $d\big(\boldsymbol{x}, \mathcal{X}_{\mathcal{W},\varepsilon}\big) \leq (\bar{l} - \underline{l})$, the following choice for γ_2 lets inequality (25) hold for all $k = 0,1, \ldots$: $\gamma_2 = \frac{\delta\varepsilon\sigma\underline{\mu}}{\bar{\eta}(\bar{l}-\underline{l})}$. ■

Remark 1. In the control law (7), $\sigma_{pq}^i[k]$ can be interpreted as the control gain, and the interpretation of Theorem 2 is that it sets an upper-bound σ on the control gain, with the twofold objective of keeping the dynamics feasible and of driving the system trajectories towards a neighborhood of $\boldsymbol{x}_{\mathcal{W}}$. ■

C. Implementation Considerations

If the selected latency functions are increasing with limited slope (see Assumption 2), the implementation of the control law (7)-(8) requires the determination of the parameters appearing in Assumption 2.e): the number of providers $|V|$ is a scenario parameter; the parameter $\bar{\lambda}$ can be set by practical knowledge either of the maximum offered load in the considered use case or of the maximum load which can be managed by the set of available providers; the sampling period δ and the maximum tolerated latency mismatch ε are set according to a practical trade-off between tight control (small values of δ and ε) and traffic overhead/convergence time (small values of δ imply more latency measures, whereas the maximum control gain σ increases with ε); the determination of the maximum latency value $\bar{l}$ and of the maximum Lipschitz constant $\bar{\eta}$ is less straightforward and is explained in the following.

In practice, it is usually simple to find a suitable latency function representing the provider performances, since they typically degrade with the load (in fact, the typical load-response time curve $f(x)$ is often modelled in the theory of M/M/1 queues by monotonically increasing functions such as as $f(x) = \frac{1}{\lambda - x}$ or $f(x) = \frac{x}{\lambda - x}$). In most cases, the latency values represent an actual measure of the performance of the providers and can be upper-bounded by a value $\bar{l}$, set according to realistic considerations. The latency value is then set equal to $\bar{l}$ whenever the value of the latency computed from the provider performance measures is larger than $\bar{l}$[1]. For instance, in the scenario described in the following Section, the latency represents the controller response time; in practice, there are

[1] In this way, Assumption 2.b) does not hold for $x \geq \bar{x}$, where $\bar{x}$ is such that $l(\bar{x}) = \bar{l}$, and, according to Property 1, there could be non-unique Wardrop equilibria with $l_p\big(x_{\mathcal{W},p}\big) = \bar{l}$ for at least one provider $p \in V$. However, this case is not interesting since it corresponds to congestion scenarios, where the provider resources are not sufficient to handle the offered load.

Quality-of-Service constraints which should be met by the provider, in terms of maximum response time, defining the upper-bound $\bar{l}$.

The maximum Lipschitz constant $\bar{\eta}$ of the latency functions, instead, must be inferred from actual provider performance measures by estimating the maximum slope of the measured latency curves. We note that, by limiting the maximum value $\bar{l}$ of the latency functions according to practical consideration (as described above), we also limit the Lipschitz constant $\bar{\eta}$ since, usually, the slope of the latency functions increases with the load, with beneficial effects on the control effectiveness since the gain σ is increased (see Assumption 2.e)).

IV. Proof-of-Concept Application to Software Defined Networking

In this section, we show how the presented algorithm has been applied to enforce the discussed load balancing mechanisms onto the distributed SDN Control Plane developed within the T-NOVA project. Focusing on the emulation of the OpenFlow control traffic rather than on the emulation of the data plane traffic, we present the results obtained from a proof-of-concept experimental setup where the considered SDN works in Equal Interaction configuration across a cluster of three OpenDaylight SDN Controllers (Lithium release).

TABLE I
Wardrop Algorithm Implementation

At the start of the control round k:

- Each SDN Proxy $i = 1, \dots, |C|$:
  - For each SDN Controller $p = 1, \dots, |V|$:
    - It computes the measured latency $l_p^{meas}[k]$ by averaging, over the last period δ, the measures of the delays between the transmission of a request to the SDN Controller p and its response. If no request was sent to the SDN Controller p in the last round, it sends a fake request to obtain the response time measure.
    - It updates the value of the latency functions by following a simple exponential averaging approach:

      $l_p[k] \leftarrow \alpha l_p[k-1] + (1-\alpha) l_p^{meas}[k]$, with $\alpha \in (0,1)$.

  - It computes the migration rates $r_{pq}^i[k], \forall p, q, \in V$ according to equation (7), with $\sigma_{pq}^i[k] = \sigma$ as defined in Assumption 2.e).
  - It computes the flow rates $x_p^i[k], \forall p \in V$, according to equation (6).

During the control round k of duration δ:

- Each SDN Proxy $i = 1, \dots, |C|$:
  - It sends the requests received from its associated SDN Switches during round k to the SDN Controllers according to a weighted round-robin scheduling, with weights proportional to the flow rates $x_p^i[k], p = 1, \dots, |V|$.

To validate the effectiveness of the proposed algorithm, we performed several tests relying on the WCBench (*Wrapped Collective Benchmark*) tool [31], which works as a generator of OpenFlow control traffic by emulating OpenFlow switches. The requests were generated on the data plane and then sent to the SDN Proxies. Each implemented SDN Proxy, developed in Python, is a network proxy in charge of catching the generated OpenFlow traffic and embeds the proposed load balancing algorithm and two other ones for comparison purposes.

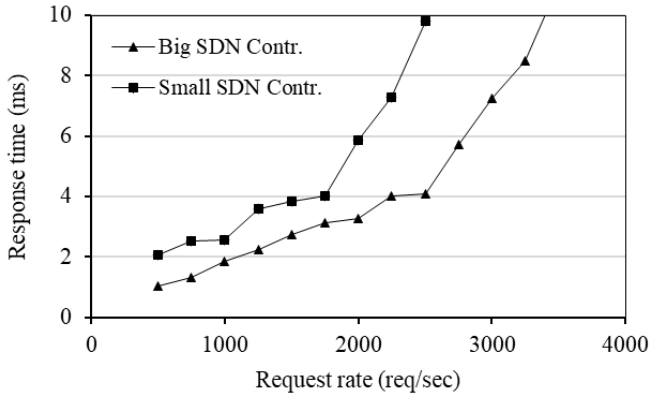

Fig. 2. Load vs. response time curves of the two considered configurations for the SDN Controller resources.

Each SDN Proxy is connected to all the available SDN Controller instances and performs a per-request balancing policy. In this case study, the latency associated with an SDN Controller is its average response time. The response time grows with the controller load and thus (i) it is a reliable indicator of the controller congestion status, (ii) it is a non-decreasing function of the request rate and therefore a suitable latency function, and (iii) it can be easily computed by the SDN Proxies, as explained below.

The time-scale is divided into rounds of duration δ. At every round, the latency is evaluated and the control actions are implemented as described in Table I.

To simulate the SDN Controller load, OpenFlow control traffic was generated on the data plane side at different request rates. The three deployed SDN Controllers were assumed to have different processing capabilities. Two configurations are used for the SDN Controller resources, namely, the *Small* one with 2GB of RAM and 2 vCPUs, and the *Big* one with 4GB of RAM and 4 vCPUs. Consequently, as shown in Fig. 2, the load-response time curves are different. The curves were obtained by measuring the average response time of the providers, denoted with $\tau_p(x_p)$, for different constant load values $x_p[k] = x_p$. In detail, a constant rate of x_p requests per second was sent for 180s to the each controller $p \in V$; the values of the x_p's ranged from $\underline{x} = 500$ req/s to the maximum load value such that the response time exceeds $\bar{l}$, denoted with $\bar{x}_p$, with granularity $\Delta x = 250$ req/s. The measured response time variations were then interpreted as latency variations, and used to estimate the maximum latency slope as $\bar{\eta} \approx \max_{p \in V, x_p \in \{\underline{x}, \underline{x}+\Delta x, \dots, \bar{x}-\Delta x\}} \frac{\tau_p(x_p+\Delta x)-\tau_p(x_p)}{\Delta x}$. From Fig. 2, we can empirically assess that the latency of the adopted SDN Controllers grows with the request rate, thus exhibiting a positive and non-decreasing behaviour that satisfies Assumption 2.

In the implemented Wardrop load balancing algorithm, the migration policy (8) is defined as:

$$\mu_{pq}^i(l_p, l_q) = \begin{cases} 0, & \text{if } l_p \le l_q + \varepsilon, \\ \min\left(\frac{l_p - l_q}{\bar{l} - \underline{l}}, \bar{\mu}\right), & \text{otherwise,} \end{cases} \forall p, q \in V, \forall i \in C.$$

The scenario and algorithm parameters are reported in Table II.

TABLE II
SCENARIO AND ALGORITHM PARAMETERS

Number of SDN Controllers	$\|V\| = 3$
Number of SDN Proxies	$\|C\| = 2$
Maximum latency value	$\bar{l} = 10\ [ms]$
Minimum latency value	$\underline{l} = 0\ [ms]$
Maximum load	$\bar{\lambda} = 10^4\ [req]$
Maximum Lipschitz constant of the latency functions	$\bar{\eta} = 4 \cdot 10^{-5}\ \left[\frac{ms^2}{req}\right]$
Latency tolerance	$\varepsilon = 0.3\ [ms]$
Maximum value of the migration policy	$\bar{\mu} = 1$
Sampling time	$\delta = 1\ [s]$
Latency averaging constant	$\alpha = 0.95$

We have compared the performance of the proposed per-request Wardrop load balancing algorithm with two approaches:

- a simple *Least Latency* (LL) load balancing algorithm, devised in a closed-loop fashion;
- an open-loop *Round Robin* (RR) load balancing algorithm, as adopted by HAProxy[2] (*High Availability Proxy*) [32].

During the k-th control round, the SDN Proxies operating with the LL algorithm send the requests received from their associated SDN Switches to the SDN Controllers according to a weighted round-robin scheduling, with weights proportional to the inverse of the values of the latency functions at time k; the latency values are updated according to an exponential averaging approach (as with the Wardrop algorithm, see Table 1). The RR algorithm, given the list of available SDN Controllers, forwards the control traffic to each SDN Controller in turn (i.e., it migrates a predefined amount of traffic to the first listed SDN Controller at time k, then to the second listed SDN Controller at time $k + 1$, and so on).

Figs. 3-6 show the test results, in terms of *throughput*, defined as the rate at which the incoming requests are processed by the SDN Control Plane, and *latency* (response time). The reported values are averaged over 20 simulation runs, each one with duration of $100s$.

The test results show that the Wardrop load balancer outperforms the other two approaches, by yielding a performance improvement evaluated in terms of total throughput and average latency (Figs. 3 and 5) of 13% and 18%, respectively, if compared with the LL algorithm, and of 17% of and 31%, respectively, if compared with the RR one.

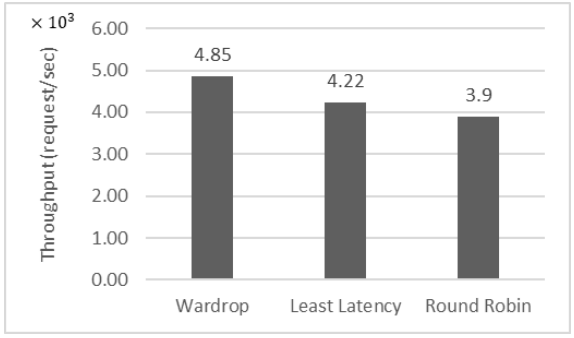

Fig. 3. Average throughput comparison among *Wardrop*, *Least Latency* and *HAProxy Round Robin* load balancing algorithms.

Fig. 5. Response time (latency) comparison among *Wardrop*, *Least Latency* and *HAProxy Round Robin* load balancing algorithms.

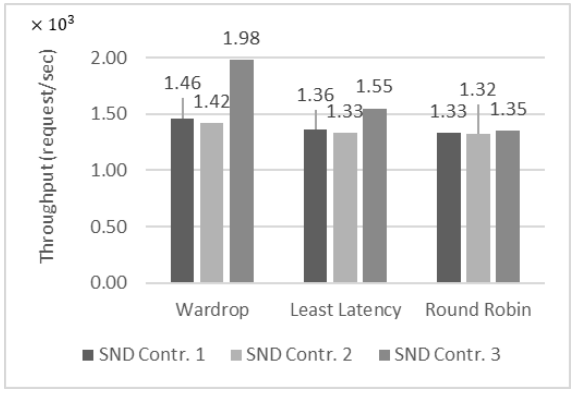

Fig. 4. Per SDN Controller throughput comparison among *Wardrop*, *Least Latency* and *HAProxy Round Robin* load balancing algorithms.

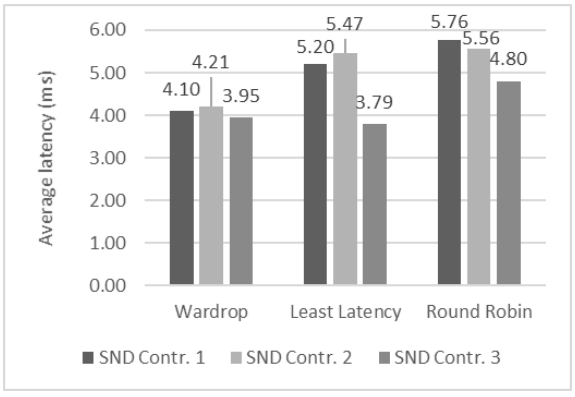

Fig. 6. Per SDN Controller response time (latency) comparison among *Wardrop*, *Least Latency* and *HAProxy Round Robin* load balancing algorithms.

[2] HAProxy is a very popular open-source software, playing the role of a TCP/HTTP Load Balancer and proxying solution, whose common use in many high-profile context (e.g., GitHub, Instagram, and Twitter) is that of improving the performance and reliability of a server environment by distributing the workload across multiple servers.

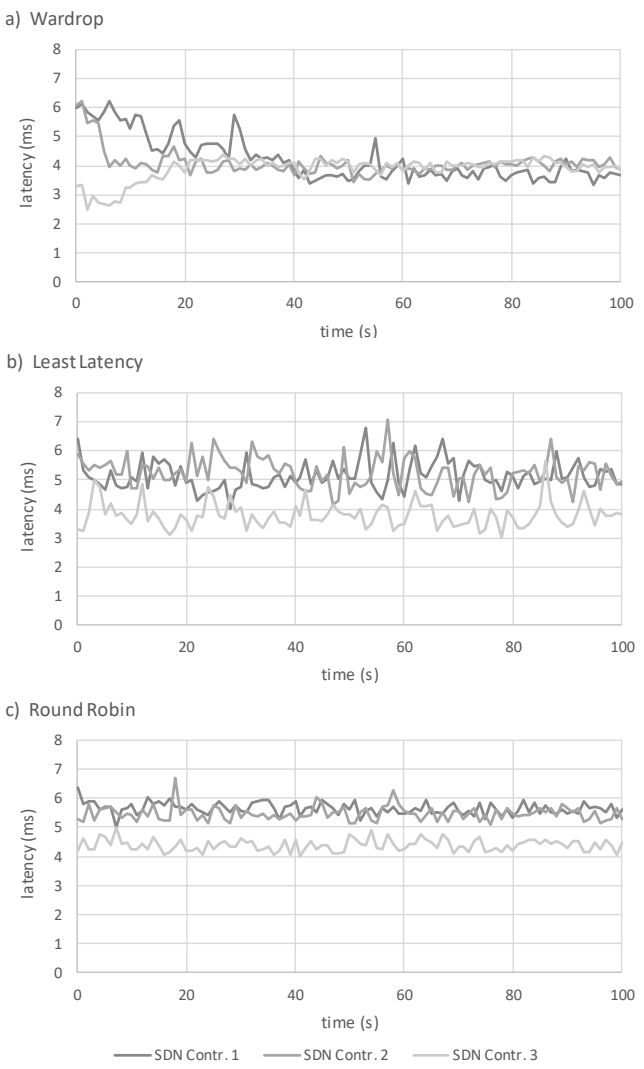

Fig. 7. Example of latency dynamics under a) *Wardrop*, b) *Least Latency* and c) *HAProxy Round Robin* load balancing algorithms.

Fig. 6 shows that the ε-Wardrop equilibrium is practically reached by the Wardrop algorithm, since the maximum difference between latencies is equal to 0.26ms; the difference grows with the LL and RR algorithms to 1.68ms and 0.96ms, respectively.

Figs. 4 and 6 show the throughput and the latency values for the three SDN Controllers and for the three algorithms. The proposed algorithm sends a larger request rate to the third SDN Controller, which has more resources (*Big* configuration) with respect to the first and second ones (*Small* configurations), in such a way that the latencies are equalized. The LL algorithm operates in a similar way but it is less effective. The RR algorithm simply balances the request rate among the three SDN Controllers, causing the latencies of the first and second SDN Controllers to grow above the latency of the third one; in practice, the third SDN Controller ends up being used less than the other two.

To make the behaviour of the algorithms clear, Fig. 7 shows examples of the transient behavior of the measured latency values under the three algorithms over a time horizon of $100s$. Figure 7.c) highlights that the open-loop RR algorithm distributes the requests in a static fashion, resulting in the larger average latencies with respect to the other closed-loop algorithms, which dynamically distribute the load based on the measured latencies. The LL algorithm (Fig. 7.b)) acts by distributing the requests proportionally to the inverse of the measured latencies, but this does not correspond to perfectly balanced latency values – in fact, the latency of the *Big* SDN controller is almost always lower than the ones of the *Small* SDN Controllers –. Moreover, the figure shows that the LL control law, being a heuristic one, causes oscillations in the latency values. Instead, Fig. 7.a) confirms that the Wardrop algorithm manages to let the latency values of the SDN Controllers converge and that it assures the lowest latencies among the three algorithms.

V. Conclusions

This paper proposes a Wardrop load balancing algorithm for SDN networks, and introduces two innovations. From the methodological viewpoint, a distributed, non-cooperative discrete-time load balancing algorithm, based on mean field game theory, is presented and is proved to converge to an arbitrarily small neighborhood of a Wardrop equilibrium. From an architectural point of view, SDN Proxies for the OpenFlow traffic are introduced to improve the scalability of SDN networks by dynamically dispatching the control workload across the available SDN Controllers. To evaluate the effectiveness of the proposed approach, a proof-of-concept implementation on a real Software Defined Network has been carried out and the related performance test results are reported. The proposed approach is scalable, since no communications among the switches is needed and no centralized load balancing algorithm must be executed by the SDN Controllers.

Future work is aimed, on the one hand, at improving the convergence properties of the algorithm, e.g., by resorting to an exact convergence to the Wardrop equilibrium (instead of the here considered set convergence) and, on the other hand, at validating the algorithm on larger use cases.

Appendix

A. Geometrical Considerations Proving Inequality (12)

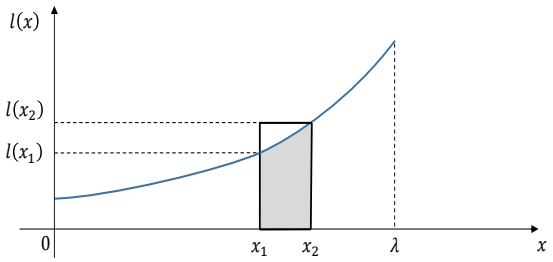

Fig. 8. Geometrical considerations proving inequality (12).

The quantity $(x_2 - x_1)l(x_2)$, represented by the area of the rectangle with bold lines in Fig. 8, is larger than the integral $\int_{x_1}^{x_2} l(s)ds$, represented by the grey area.

Considering the areas $\mathcal{B}_q$ and $\mathcal{A}_q$ depicted in Fig. 9, we can write:

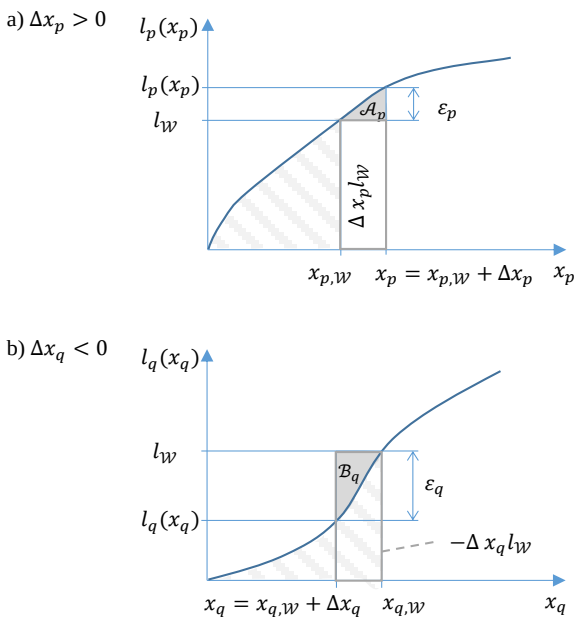

Fig. 9. Geometrical considerations proving argument B1 in Section III.B.

$$\Phi(\boldsymbol{x}) - \Phi_{min} = \sum_{p\in V|\Delta x_p>0}\left(\Delta x_p l_W + \mathcal{A}_p\right) - \sum_{q\in V|\Delta x_q<0}\left(\Delta x_q l_W - \mathcal{B}_q\right) = \sum_{p\in V|\Delta x_p>0}\mathcal{A}_p + \sum_{q\in V|\Delta x_q<0}\mathcal{B}_q \geq \sum_{p\in V|\Delta x_p>0}\frac{\varepsilon_p^2}{2\bar{\eta}} + \sum_{q\in V|\Delta x_q<0}\frac{\varepsilon_q^2}{2\bar{\eta}},$$

where the last equality holds since $\mathcal{A}_p \geq \frac{\varepsilon_p^2}{2\bar{\eta}}$ and $\mathcal{B}_q \geq \frac{\varepsilon_q^2}{2\bar{\eta}}$ and where $\varepsilon_p \coloneqq l_p(x_p) - l_W$ and $\varepsilon_q \coloneqq l_W - l_q(x_q)$.

Since $\boldsymbol{x} \in \mathcal{X} \setminus \mathcal{X}_{W,\varepsilon}$, there exists at least a couple $p', q' \in V$ such that $\Delta x_{p'} > 0$ $\Delta x_{q'} < 0$ and $\varepsilon_{p'} + \varepsilon_{q'} > \varepsilon$. It follows that:

$$\Phi(\boldsymbol{x}) - \Phi_{min} \geq \frac{\varepsilon_{p'}^2}{2\bar{\eta}} + \frac{\varepsilon_{q'}^2}{2\bar{\eta}} \geq \frac{\left(\varepsilon_{p'}+\varepsilon_{q'}\right)^2}{4\bar{\eta}} > \frac{\varepsilon^2}{4\bar{\eta}}.$$

ACKNOWLEDGMENT

The authors wish to thank Prof. R. Baldoni and all the members of the T-NOVA project.

REFERENCES

[1] "Software-defined networking: The new norm for networks," *Open Networking Foundation*, 2012, https://www.opennetworking.org
[2] A. Dixit, F. Hao, S. Mukherjee, T.V. Lakshman, R. Kompella, "Towards an Elastic Distributed SDN Controller," *ACM SIGCOMM Computer Communication Review*, Volume 4, 2013.
[3] D. Grosu, A. T. Chronopoulos, and M. Y. Leung, "Cooperative load balancing in distributed systems," *Concurr. Comput. Pract. Exp.*, vol. 20, no. 16, pp. 1953–1976, Nov. 2008.
[4] S. U. Khan and I. Ahmad, "A Cooperative Game Theoretical Technique for Joint Optimization of Energy Consumption and Response Time in Computational Grids," *IEEE Trans. Parallel Distrib. Syst.*, vol. 20, no. 3, pp. 346–360, Mar. 2009.
[5] R. Subrata, A. Y. Zomaya, and B. Landfeldt, "A Cooperative Game Framework for QoS Guided Job Allocation Schemes in Grids," *IEEE Trans. Comput.*, vol. 57, no. 10, pp. 1413–1422, Oct. 2008.
[6] D. Grosu and A. T. Chronopoulos, "Noncooperative load balancing in distributed systems," *J. Parallel Distrib. Comput.*, vol. 65, no. 9, pp. 1022–1034, Sep. 2005.
[7] R. Subrata, A. Y. Zomaya, and B. Landfeldt, "Game-Theoretic Approach for Load Balancing in Computational Grids," *IEEE Trans. Parallel Distrib. Syst.*, vol. 19, no. 1, pp. 66–76, Jan. 2008.
[8] E. Even-Dar, A. Kesselman, and Y. Mansour, *Convergence time to Nash equilibria*. Springer-Verlag, Berlin Heidelberg, 2003.
[9] E. Even-Dar and Y. Mansour, "Fast convergence of selfish rerouting," in *16th annual ACM-SIAM symposium on Discrete algorithms, Society for Industrial and Applied Mathematics*, 2005, pp. 772–781.
[10] S. Shah and R. Kothari, "Convergence of the dynamic load balancing problem to Nash equilibrium using distributed local interactions," *Inf. Sci. (Ny).*, vol. 221, pp. 297–305, Feb. 2013.
[11] H. Ackermann, S. Fischer, M. Hoefer, and M. Schöngens, "Distributed algorithms for QoS load balancing," *Distrib. Comput.*, vol. 23, no. 5–6, pp. 321–330, Dec. 2010.
[12] J. G. Wardrop, "Some Theoretical Aspects of Road Traffic Research," *ICE Proc. Eng. Div.*, vol. 1, no. 3, pp. 325–362, Jan. 1952.
[13] H. Kameda, J. Li, C. Kim, and Y. Zhang, *Optimal Load Balancing in Distributed Computer Systems*. London: Springer London, 1997.
[14] S. Fischer, H. Räcke, and B. Vöcking, "Fast Convergence to Wardrop Equilibria by Adaptive Sampling Methods," *SIAM J. Comput.*, vol. 39, no. 8, pp. 3700–3735, Jan. 2010.
[15] D. Barth, O. Bournez, O. Boussaton, and J. Cohen, "Distributed learning of Wardrop equilibria," in *Lecture Notes in Computer Science, vol. 5204*, 2008, pp. 19–32.
[16] G. Oddi and A. Pietrabissa, "A distributed multi-path algorithm for wireless ad-hoc networks based on Wardrop routing," in *21st Mediterranean Conference on Control and Automation*, 2013, pp. 930–935.
[17] S. Battilotti, F. Delli Priscoli, C. Gori Giorgi, M. Panfili, A. Pietrabissa, L. Ricciardi Celsi, and V. Suraci, "A Multi-Agent Reinforcement Learning Based Approach to Quality of Experience Control in Future Internet Networks," in *34th IEEE Chinese Control Conference (CCC2015)*, 2015, pp. 6495–6500.
[18] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: enabling innovation in campus networks," *SIGCOMM Comput. Commun. Rev.*, vol. 38, pp. 69-74, Mar. 2008.
[19] Kandula et al., "The nature of data center traffic: measurements & analysis," in *Proceedings of IMC 2009*, ACM, pp. 202-208, 2009.
[20] L. Zuccaro, F. Cimorelli, F. Delli Priscoli, C. Gori Giorgi, S. Monaco, V. Suraci, "Distributed Control in Virtualized Networks," in *Proc. Of the 10th International Conference on Future Networks and Communications (FNC 2015)*, pp. 276-283, 2015.
[21] Y.-N. Hu, W.-D. Wang, X.-Y. Gong, X.-R. Que, and S.-D. Cheng, "On the placement of controllers in software-defined networks," *J. China Univ. Posts Telecommun.*, vol. 19, no. S2, pp. 92–171, Oct. 2012.
[22] S. Lange et al. "Heuristic Approaches to the Controller Placement Problem in Large Scale SDN Networks", *IEEE Transactions on Network and Service Management*, Year: 2015, Volume: 12, Issue: 1 Pages: 4 - 17, DOI: 10.1109/TNSM.2015.2402432.
[23] A. Sallahi, M. St-Hilaire, "Optimal Model for the Controller Placement Problem in Software Defined Networks", *IEEE Communications letters*, vol. 19, no. 1, january 2015. DOI: 10.1109/LCOMM.2014.2371014.
[24] F. Cimorelli, F. Delli Priscoli, A. Pietrabissa, L. Ricciardi Celsi, V. Suraci, and L. Zuccaro, "A Distributed Load Balancing Algorithm for the Control Plane in Software Defined Networking," in *Proceedings of the 24th Mediterranean Conference on Control and Automation (MED 2016)*, pp. 1033-1040, June 21-24, 2016, Athens, Greece, DOI: 10.1109/MED.2016.7535946.
[25] J. Medved et al., "OpenDaylight: Towards a model-driven SDN controller architecture," in *2014 IEEE 15th International Symposium on a World of Wireless, Mobile and Multimedia Networks*, 2014.
[26] G. Oddi, A. Pietrabissa, F. Delli Priscoli, F. Facchinei, L. Palagi, A. Lanna, "A QoE-aware dynamic bandwidth allocation algorithm based on game theory," in *Proceedings of the 23rd Mediterranean Conference on Control and Automation (MED 2015)*, 2015, pp. 979-985.
[27] M. Beckmann, C.B. McGuire, C.B. Winsten, *Studies in the Economics of Transportation*. Yale University Press, New Haven, CT, 1956.
[28] T. Roughgarden and E. Tardos, "How bad is selfish routing?," *J. ACM*, 49(2): 236–259, 2002.
[29] S. Fischer, B. Vöcking, "Adaptive routing with stale information," *Theoretical Computer Science*, vol. 410, no. 36, 2009, pp. 3357-3371.
[30] G. A. Shanholt, "Set stability for difference equations," *International Journal of Control*, vol. 19, no. 2, pp. 309-314, 1974.
[31] https://github.com/dfarrell07/wcbench
[32] http://www.haproxy.org/

Antonio Pietrabissa is Assistant Professor at the Department of Computer, Control, and Management Engineering "Antonio Ruberti" (DIAG) of the University of Rome "La Sapienza," where he received his degree in Electronics Engineering and his Ph.D. degree in Systems Engineering in 2000 and 2004, respectively, and where, since 2010, he has been teaching "Automatic Control" and "Process Automation". Since 2000, he has participated (either as scientific responsible or as technical coordinator) in about 20 EU and National research projects. His main research focus is on the application of systems and control theory methodologies to the analysis and control of communication networks, with specific interest to the design of resource management protocols. He is author of about 40 journal papers and 80 conference papers and book chapters on these topics.

Lorenzo Ricciardi Celsi was born in Rome, Italy, in 1990. He received the B.Sc. degree in Electronics Engineering in 2011 and the M.Sc. degree in Control Engineering in 2014, both *summa cum laude* from the University of Rome "La Sapienza." He also received the PhD degree in Automatica, Bioengineering and Operations Research from the same university and the PhD degree in Sciences et Technologies de l'Information et de la Communication, Specialité Automatique, from Université Paris-Saclay in 2018, both *cum laude*. He is currently a Postdoctoral Research Fellow at the Department of Computer, Control, and Management Engineering Antonio Ruberti (DIAG) of the University of Rome La Sapienza. He has been working on reinforcement learning algorithms within the framework of the FP7 project T-NOVA and the MIUR project PLATINO. He is also taking part in the H2020 projects BONVOYAGE and ATENA. His main research interests are: nonlinear networked systems and control in general, cooperative control methodologies for multi-agent systems, and cyber-physical security of critical infrastructures.

Federico Cimorelli was born in 1989, in Venafro, Italy. He received the degree in Engineering in Computer Science in 2014 from the University of Rome "La Sapienza." He also received, in 2018, the PhD degree in Automatica, Bioengineering and Operations Research from the same university, where he is now a Postdoctoral Research Fellow. Since 2014, he has been also with the Consortium for Research in Automation and Telecommunications (CRAT), involved in both Italian and European Research Programs in the ICT field. His main research topics include the application of systems theory to recent problems in the fields of Software Defined Networking (SDN) and Network Function Virtualization (NFV).

Vincenzo Suraci was born in Rome, Italy, in 1978. He graduated in Computer Engineering *summa cum laude* in 2004 at the University of Rome "La Sapienza." In 2008 he pursued a

Ph.D. degree in Systems Engineering at the Department Computer, Control, and Management Engineering "Antonio Ruberti" (DIAG) of the same university. Currently, he is an Associate Professor at eCampus University and Project Manager at CRAT. His main research interest is to develop and adapt advanced control and operations research methodologies (such as reinforcement learning, column generation, hybrid automata, and discrete event systems) for the solution of challenging and emerging engineering problems: e.g., connection admission control, access technologies selection, QoE/QoS cognitive control, resource management over heterogeneous technologies, convergence of heterogeneous networks. He has achieved a wide experience in the field of applied research and project management. Since 2011, he has been managing the EU-funded Future Internet Core Platform research project FI-WARE. In 2012, he also applied for a EU Patent request on DVB as a result of his profitable research in the framework of EU research projects.

Francesco Delli Priscoli was born in Rome, Italy, in 1962. He received the degree in Electronics Engineering (*summa cum laude*) and the Ph.D. in Systems Engineering from the University of Rome "La Sapienza" in 1986 and 1991, respectively. From 1986 to 1991 he worked in Telespazio. Since 1991, he has been working at the University of Rome "La Sapienza," where, at present, he is Full Professor of "Automatic Control," "Control of Autonomous Multi-Agent Systems," and "Control of Communication and Energy Networks". In the framework of his academic activity, he has mainly researched on resource/service/content management procedures and on cognitive techniques for telecommunication and energy networks, by largely adopting control-based methodologies. He is the author of about 180 papers appeared in major international journals (about 60), on books (about 10) and in conference proceedings (about 110). He also holds four patents. He is an associate editor of Control Engineering Practice and a member of the IFAC Technical Committee on "Networked Systems". He was/is the scientific responsible, at the University of Rome "La Sapienza", for 31 projects financed by the European Union (Fourth, Fifth, Sixth, Seventh and Eighth Framework Programmes) and by the European Space Agency (ESA). His present research interests concern closed-loop multi-agent learning techniques for Quality of Experience evaluation and control in advanced communication and energy networks, as well as all the related networking algorithms.

Alessandro Di Giorgio was born in Rome, Italy, in 1980. He received the degree (*cum laude*) in Physics in 2005, and the Ph.D. degree in Systems Engineering from the University of Rome "La Sapienza," in 2010. He is currently a Research Fellow in Automatic Control, working on original applications of control systems theory to the resource management problem in the field of power systems and telecommunication networks; he is

author of about 40 papers and book chapters on these topics, mainly produced in the context of national and European research projects.

Alessandro Giuseppi was born in Rome, Italy, in 1992. He received from the University of Rome "La Sapienza" his B.Sc. degree in Computer and Automation Engineering in 2014 and his M.Sc. degree in Control Engineering in 2016, both *summa cum laude*. Currently, he is a PhD Candidate in Automatica, Bioengineering and Operations Research at the same university. He started his research activities in the FP7 project T-NOVA, where he studied Reinforcement Learning applications to Software Defined Networks. He is currently working as a researcher in the H2020 project ATENA, dealing with Critical Infrastructure Security. His other research activities and interests are Model Predictive Control, Neural Networks and their applications to security-related issues in cyber-physical systems.

Salvatore Monaco was born in 1951 and he has been a Full Professor of Systems Theory at the University of Rome "La Sapienza" since 1986. He was a member of the ASI (Italian Space Agency) Scientific Committee from 1989 to 1995, of the Executive Council of the EUCA (European Union Control Association) from 1990 (foundation year) to 1997, and of the ASI Working Group on Evaluation from 1999 to 2001. He has also been a member of the ASI Technological Committee since 1997. He has promoted technological transfer in the area of Automation. In 1995, he served as scientific advisor for the Director of the Joint Research Center of the European Union. Since 2001, he has been president of the council for the degree of Systems and Control Engineering at the University of Rome "La Sapienza" and also president of the Scientific Committee of the "Université Franco-Italienne," an inter-governmental institution for coordinating research and didactics. His research activity is in the field of Systems and Control Theory and applied research in spacecraft control, mobile robot control and control of telecommunication networks.