AIMD-inspired switching control of computing networks

Vlahakis, E., Jungers, R., Athanasopoulos, N., & McLoone, S. (2023). AIMD-inspired switching control of computing networks. *IEEE Transactions on Control of Network Systems*. Advance online publication. https://doi.org/10.1109/TCNS.2023.3298202

Published in:
IEEE Transactions on Control of Network Systems

Document Version:
Peer reviewed version

AIMD-inspired switching control of computing networks

Eleftherios Vlahakis, Raphaël Jungers, Nikolaos Athanasopoulos, and Seán McLoone

***Abstract*— We consider the scheduling problem of requests entering a distributed computing network consisting of a set of non-cooperative nodes, where a node is represented by a queue combined with a computing unit. Our interaction-free setup between nodes renders decentralised scheduling challenging, with most existing results focusing on centralised or static solutions. Inspired by congestion control, we propose a new average-based additive increase multiplicative decrease (AIMD) admission control policy which requires minimal communication between individual nodes and an aggregator. The proposed admission policy infers a discrete-event model expressed as a positive constrained switching system that is triggered whenever the queue of the aggregation point of requests vanishes. We show convergence of the proposed AIMD system under unknown, peak-bounded workload profiles by analysing the spectrum of rank-one perturbations of symmetric matrices and the boundedness of the joint spectral radius of sets of symmetric matrices. Contrary to methods that address scheduling and resource allocation asynchronously or via a two-step approach, our AIMD-based scheme can tackle both tasks simultaneously. This is illustrated by proposing a decentralised resource allocation controller coupled with the scheduling scheme leading to a stable closed-loop control system, that is guaranteed to avoid underutilisation of resources and is tunable via the sets of AIMD parameters.**

***Index Terms*— AIMD, scheduling, queueing systems, discrete-event systems, event-triggered systems, constrained switching systems, state-dependent switching systems, decentralised resource allocation**

I. Introduction

Distributed computing is an important technology that is emerging to address the ever-growing demand for extensive, real-time computations at the *edge* as a result of the proliferation of end-user devices connected to the edge of the Internet. Although this emerging paradigm opens new opportunities for more sophisticated applications [1], it presents several research challenges, especially in the context of resource allocation and control due to latency constraints, limited capacity of edge-servers, and a decentralized and volatile structure.

In this paper, we consider the joint problem of scheduling and provisioning resources when multiple requests are offloaded to a system of distributed computing nodes. We follow a queueing system approach to modelling that is simple, scalable, and agnostic to each individual node's specificities. Arguably, analytical modelling of computer systems is challenging [2]–[4], with many works relying on application-specific models obtained via system identification methods [5], [6]. To model local backlog, we associate each node with a queue combined with a computing unit. We assume no direct interaction between nodes and that computing units are independent of one another. Nodes are only coupled through a central node, which is key to the proposed modelling and control approach. We note that such a structural requirement is not necessarily restrictive but, in fact, practical as it permits requests to be dispatched from a source acting as aggregation point [7]–[10].

Scheduling and load balancing algorithms are typically static or dynamic. Static methods spread the workload over a set of computing nodes guided by rules set a priori. The workload sharing is formulated as an optimisation program [11] or a static network flow control problem [12]. Optimisers are obtained with respect to an appropriate cost function typically under *worst-case* workload parameters. Static schemes fail to adapt to unforeseen workload variations. Dynamic methods, instead, compensate for this drawback at the expense of a more complex structure. Although open-loop approaches can operate reliably for well-modelled workloads [7], feedback-based methods are required for highly volatile request flows [9]. Continuous monitoring of the entire system's state for feedback actions, however, may be challenging, rendering the study of more efficient schemes necessary for scheduling control. This challenge motivates our focus on the additive increase multiplicative decrease (AIMD) algorithm that fits an event-triggered feedback mechanism with decentralised structure.

Our approach to scheduling is inspired by the AIMD algorithm, a celebrated method in network management. The AIMD algorithm was originally introduced in [13] for tackling, in a robust and decentralized manner, congestion in computer networks requiring minimum interaction between nodes. It has become a fundamental building block of the Transmission Control Protocol (TCP) widely used across the Internet. An

This work was supported by the CHIST-ERA grant CHIST-ERA-18-SDCDN-003 (DRUID-NET). R. Jungers is supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme under grant agreement No 864017-L2C.

E. Vlahakis is with the Division of Decision and Control Systems, School of Electrical Engineering and Computer Science, KTH Royal Institute of Technology, Stockholm, Sweden. R. Jungers is with ICTEAM institute, UCLouvain, Louvain-la-Neuve, Belgium. N. Athanasopoulos and S. McLoone are with School of Electronics, Electrical Engineering and Computer Science, Queen's University Belfast, Northern Ireland, UK. Email: vlahakis@kth.se, raphael.jungers@uclouvain.be, {n.athanasopoulos, s.mcloone}@qub.ac.uk.

excellent and comprehensive study of the AIMD algorithm with several extensions and applications can be found in [14].

In this paper, we first attempt to address the question: Is the AIMD algorithm compatible with request scheduling and load balancing tasks where the shared quantity should, on average, be distributed in full and not in part as in a congestion-avoidance setup [15], [16]? Roughly, a typical AIMD scheme relates to resource sharing among self-organised nodes, each increasing their share until the resource is exhausted (see, e.g., [17], [18]). When this happens, a one-bit signal is fed to nodes triggering an instantaneous decrease in their share. The event-wise dynamics of this increase-decrease pattern induce an event-driven discrete model. Stability of this system is guaranteed, in the sense that if the resource capacity is constant then individual shares converge. It is worth noting that under this scheme the shared quantity is not fully utilised on average. Consequently, the scheme cannot straightforwardly be applied to workload-sharing tasks, because in such a case the backlog would grow indefinitely.

To address this challenge, we propose a new triggering condition that is compatible with the AIMD logic, and introduce a novel AIMD-inspired control algorithm for general scheduling tasks. Berman *et al.* in [19] and Shorten *et al.* in [15] show that a typical AIMD scheme can be modelled at event times as a positive system, thus, convergence properties can be inferred from the Perron–Frobenius theorem [14]. Unfortunately, such properties do not hold here due to a new event-triggered mechanism. Instead of capacity constraints, we consider an event-based mechanism triggered when the backlog corresponding to the central node of our model vanishes. To the best of our knowledge, this constitutes a novel triggering condition for scheduling tasks in the context of distributed computing and discrete-event systems. By embedding the triggering condition into the dynamics, we derive a closed form of the admission control system expressed as a constrained switching system. Our convergence analysis follows a set-theoretic approach and relies on a fundamental result in Linear Algebra [20], [21] involving the eigenproblem of rank-one perturbations of symmetric matrices (Theorems 4.1 and 4.2), and the boundedness of the joint spectral radius of symmetric matrices [22] (Theorem 4.3 and Proposition 4.1). The admission policy is locally configurable and convergent irrespective of tuning and system dimension (number of computing nodes). It also inherits the fairness feature (see Remark 10) of the standard AIMD algorithm that is obtained by tuning local parameters [16].

As a result of the simplicity of the proposed AIMD scheduling policy, we subsequently formulate a resource allocation strategy defined as a decentralized, stabilising nonlinear feedback controller. Under the proposed resource allocation law, individual queues (local backlogs) are bounded, and converge to sets of values in finite time. This effectively permits analysis of Quality of Service (QoS) metrics, e.g., queueing time. Overall, scheduling and resource allocation lie in the same control loop leading to a simple decentralized system that is stable, scalable, and locally configurable. A key asset of our approach is that the closed-loop dynamics of the overall system can be tuned by means of two sets of parameters, namely the AIMD parameters. Centralised algorithms for optimal AIMD tuning that are in agreement with the schemes presented in this paper can be found in [23]. The authors are currently studying decentralised configurations for efficient AIMD tuning.

Complementing stochastic methods [24], [25], we consider non-deterministic workloads that are unknown, yet bounded. Under this assumption, deterministic stability and convergence properties can be derived using set-theoretic methods [26]. We show that the admission control system under unknown, bounded workload profiles always converges to a compact set that is invariant and reachable in finite time irrespective of the initial conditions. Under the proposed resource allocation strategy, boundedness is also guaranteed for the local backlog under bounded time-varying workload. Our control strategy has successfully been applied to *Kubernetes*[1] under a realistic workload profile outperforming state-of-the-art scheduling solutions and demonstrating a better resource utilisation by reducing computing resources by 8%, without any prior tuning of the related parameters [10].

Preliminary results of this paper have been presented in [27] under the assumption of constant workload. Here, we show that a volatile workload profile requires a constrained switching system modelling approach that is significantly more complex. This effectively guarantees that the underlying system is positive. This approach, which involves convergence analysis in the context of piecewise affine systems, is not considered in [27].

Our contributions are summarized as follows.

- We propose a new AIMD-based scheduling solution that guarantees bounded backlog, allowing self-organised computing nodes to address time-varying workload profiles dynamically. A novel event-triggered mechanism is introduced that requires minimal communication with an aggregation point.
- We provide formal guarantees of convergence via set-theoretic techniques that involve reachability analysis of a piecewise affine system and differ substantially from typical positive-system approaches employed in the analysis of AIMD algorithms. These results can be used in a straightforward fashion to derive deterministic bounds on important QoS metrics.
- We propose a new stabilising feedback controller for decentralised resource allocation, which prevents resource over-provisioning and permits scheduling and resource allocation to be tackled simultaneously, leading to a stable closed-loop control system.

The remainder of the paper is organized as follows. Notation, definitions and assumptions are given in Section II. The problem considered in the paper is described in Section III. The main results, namely, the AIMD scheduling strategy and its convergence analysis, and the resource allocation control, are presented in Sections IV and V, respectively. A comparative study of the proposed AIMD-based scheme with standard scheduling solutions is reported in Section VI. Section VII discusses our main results and future research.

[1]Kubernetes is an open-source platform for automatic deployment, scaling, and management of containerized applications (https://kubernetes.io/).

II. Preliminaries

A. Notation

The sets of nonnegative integers and real numbers are $\mathbb{N}$ and $\mathbb{R}$, respectively, the set of n-dimensional vectors with real elements is $\mathbb{R}^n$, and the set of $n \times m$ real matrices is $\mathbb{R}^{n\times m}$. The sets of nonnegative and positive real numbers are $\mathbb{R}_{\geq 0}$ and $\mathbb{R}_{>0}$, respectively. The set of n-dimensional real vectors with (non-negative) positive elements is $(\mathbb{R}^n_{\geq 0})$ $\mathbb{R}^n_{>0}$. The transpose of a vector ξ is $\xi^\top$. Let $a_1, \ldots, a_n \in \mathbb{R}$, then, $a = (a_1, \ldots, a_n) \in \mathbb{R}^n$, and $A = \text{diag}(a_1, \ldots, a_n)$ is a diagonal matrix, where $a_1, \ldots, a_n$ are its diagonal elements. We denote by $\det(A)$ the determinant of a square matrix A. The identity matrix of dimension $m \times m$ is denoted by $I_m \in \mathbb{R}^{m\times m}$, while the vectors with elements equal to one and zero are $1 \in \mathbb{R}^n$ and $0 \in \mathbb{R}^n$, respectively. Let $\lambda_i(\Phi)$, $i = 1, \ldots, m$, be the i^{th} eigenvalue of matrix $\Phi \in \mathbb{R}^{m\times m}$. The spectrum of Φ is $\sigma(\Phi) = \{\lambda_1(\Phi), \ldots, \lambda_m(\Phi)\}$. Whenever there exists a nonsingular matrix $T \in \mathbb{R}^{m\times m}$ such that $\hat{\Phi} = T^{-1}\Phi T$ is a symmetric matrix, we assume, without loss of generality, $\lambda_1(\Phi) \leq \lambda_2(\Phi) \leq \ldots \leq \lambda_m(\Phi)$. A matrix $\Phi \in \mathbb{R}^{m\times m}$ is Schur if all its eigenvalues strictly lie inside the unit circle, i.e., $|\lambda_i(\Phi)| < 1$, $i = 1, \ldots, m$. Right matrix products of the form $A_{j_k} \ldots A_{j_1} A_{j_0} \in \mathbb{R}^{n\times n}$ are written compactly as $\prod_{i=k}^{0} A_{j_i}$, with $\prod_{i=k}^{k} A_{j_i} = A_{j_k}$, and $\prod_{i=k}^{k-1} A_{j_i} = I_n$. Let $\Sigma = \{A_1, \ldots, A_m\} \in \mathbb{R}^{n\times n}$ be a set of matrices in $\mathbb{R}^{n\times n}$. The joint spectral radius of Σ is defined as $\rho(\Sigma) = \lim_{k\to\infty} \max_{(j_1,\ldots,j_k)\in\{1,\ldots,m\}^k} \| \prod_{i=k}^{1} A_{j_i} \|^{1/k}$. In the sequel, we denote by $\rho(\Sigma)$ the spectral radius or the joint spectral radius of Σ if Σ denotes a single matrix or a finite set of matrices, respectively. The Minkowski sum of sets X_1 and X_2 is $X_1 \oplus X_2$. The convex hull of a set X is $\text{co}(X)$. Let $a, b \in \mathbb{R}_{>0}$. The remainder of the division of a by b is $\text{mod}(a, b)$.

B. Definitions and queue modelling

A *request* is an individual call for computing resources and a *computing node* is the physical or virtual computing environment whereby the content of arriving requests is processed. A *distributed computing network* (DCN) consists of multiple computing nodes. We call *workload* the arrival rate of requests and *egress* the departure rate of requests per unit time. A *queue* is the waiting mechanism whereby requests entering a node are temporarily put on hold until they are selected for service. We consider queues consistent with the *First Come First Served* (FCFS) selection principle, and requests associated with a single application. A *queueing system* [28], [29] is defined as the dynamic relationship developed between workload and egress in the presence of a queue.

We model a DCN as a multi-queue scheme whereby new requests are first queued in a central node and are subsequently dispatched to computing nodes. We refer to the queue of the central node acting as an aggregation point as the *buffer*, and to the frequency of requests being dispatched from the buffer to an individual node as the *admission rate*. Similarly, we call *service rate* the frequency of requests being processed by a node. The former is the fraction of workload assigned to an individual node while the latter is the egress of an individual node. A *local* queue is associated with an individual node. We use the term *backlog* to express queued requests either in the buffer or in local queues. A request entering a node is queued before being selected for service unless the backlog of the underlying node is zero.

III. Problem description

Throughout the paper, we use the following notation as shown in Fig. 1. We denote continuous time by $t \geq 0$, and distinct instants by $t_k \geq 0$, with $k \in \mathbb{N}$. We consider that requests associated with a specific application enter a DCN at an arrival rate $\lambda(t) \in L \subset \mathbb{R}_{>0}$. We call this the workload of the DCN associated with an application. The states of the admission system are the individual admission rates each determining the instantaneous workload share that is assigned to a node. This assignment task is called scheduling. We also wish to design a resource allocation strategy that determines individual service rates ensuring that all requests are served in finite time and backlog is bounded for all $t \geq 0$. Backlog in the buffer is denoted by $\delta(t) \in \mathbb{N}$, while the local backlog in the i^{th} computing node is denoted by $w_i(t) \in \mathbb{N}$, $i = 1, \ldots, n$. Admission and service rates associated with the i^{th} node are represented by $u_i(t) \in \mathbb{R}_{>0}$ and $\gamma_i(t) \in \mathbb{R}_{>0}$, respectively, $i = 1, \ldots, n$. Omitting the time dependency, aggregate vectors are $u = (u_1, \ldots, u_n)$, $w = (w_1, \ldots, w_n)$, $\gamma = (\gamma_1, \ldots, \gamma_n)$. An admission and resource allocation policy is denoted as $(u, \gamma) \in \mathbb{R}^{2n}$. A multi-queue scheme representing the dynamical structure of our DCN model is depicted in Fig. 1.

Remark 1: The adoption of rates for admission and resource allocation control induces a *fluid modelling approach* where system variables model the evolution of the expectation of physical quantities of the actual system. Fluid approximations are typical in management of computing systems (see, e.g., [7]).

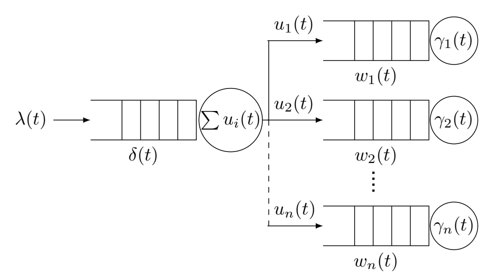

Fig. 1. A DCN as a multi-queue scheme.

A. Event-triggered mechanism

We follow a discrete-event system approach to modelling a DCN as a multi-queue scheme. An event generator is introduced as the mechanism indicating instants at which a well-defined triggering condition is satisfied. The event-wise evolution of time can be modelled as $t_{k+1} = t_k + T(k)$, where t_k denotes the time at which the k^{th} event occurs (is generated), and $T(k)$ is called the *inter-event period* and is permitted to be time-varying. The obtained discrete-event model is a result of 1) the presence of a buffer (central queue), and 2) the admission control policy.

B. Discrete-event model with AIMD dynamics

We consider a set of n computing nodes and assume that a workload $\lambda(t)$ enters the system via a central queue (buffer), as illustrated in Fig. 1. We assume that $\lambda : \mathbb{R}_{\geq 0} \to L$ is a right-continuous function, with $L \subset \mathbb{R}_{>0}$. The continuous-time dynamics of the multi-queue system with backlog $[\delta(t)\ w(t)^\top]^\top$ can be written in a compact form as

$$\begin{bmatrix} \dot{\delta}(t) \\ \dot{w}(t) \end{bmatrix} = \begin{bmatrix} 1 & -1^\top & 0^\top \\ 0 & I & -I \end{bmatrix} \begin{bmatrix} \lambda(t) \\ u(t) \\ \gamma(t) \end{bmatrix}. \tag{1}$$

We transform (1) to a discrete-event system by introducing the following triggering mechanism. An event occurs at t_k, $k \in \mathbb{N}$, when the backlog in the buffer vanishes, i.e., when

$$\delta(t_k) = 0. \tag{2}$$

Condition (2) implies that all the requests queued in the buffer up to time t_k have been dispatched to individual nodes. Hence, the buffer instantaneously vanishes at t_k. The sequence $\{t_k\}_{k=0}^{\infty}$ is non-decreasing, i.e., $t_k \leq t_{k+1}$. By associating t_k with the k^{th} event, we count events by the sequence $\{k\}_0^{\infty}$. The following scheduling logic guarantees the generation of events. Intuitively, should the buffer be non-empty, individual nodes attempt to drain the buffer by increasing their admission rates linearly. When an event occurs at t_k, individual admission rates drop instantaneously. We stress that this is the opposite of the classic AIMD scheme [13] where individual nodes attempt to fill up the underlying buffer.

Remark 2: Although the proposed triggering mechanism may reduce the communication overhead of the DCN in hand, it is chosen as it enables full workload dispatch in finite time, which is essential in our setting. Thus, the triggering condition (2) pertains to a context different from typical event-triggered transmission schemes [30], [31] whose fundamental objective is to minimise the use of communication resources by limiting unnecessary transmissions of data.

By denoting

$$u_i(t_k^-) = \lim_{\substack{t \to t_k \\ t < t_k}} u_i(t), \quad u_i(t_k^+) = \lim_{\substack{t \to t_k \\ t > t_k}} u_i(t), \tag{3}$$

i.e., the admission rate of the i^{th} node right before and right after the k^{th} event, respectively, we define

$$u_i(t_k^+) = \beta_i u_i(t_k^-), \tag{4}$$

where $0 \leq \beta_i < 1$ is called the *multiplicative decrease parameter* or *drop factor*. Intuitively, when the buffer is empty, individual admission rates are triggered to instantaneously shrink to a fraction of $u_i(t_k^-)$ according to (4). This is called the *Multiplicative Decrease* (MD) phase. We note that multiple successive drops are permitted during the MD phase. Instantaneous drops are carried out as long as the buffer remains empty. Right after the MD phase, i.e., when $\delta(t_k^+) > 0$, we require the i^{th} admission rate $u_i(t)$ grow in a ramp fashion as

$$u_i(t) = \beta_i u(t_k^-) + \alpha_i (t - t_k),\ t > t_k, \tag{5}$$

where the slope of the ramp $\alpha_i > 0$ is called the *growth rate* or the *increase parameter*. Since $u_i(t)$ is strictly increasing in t, there exists a finite $t_{k+1} \geq t_k$ such that $\delta(t_{k+1}) = 0$, under a bounded workload. We call $T(k) = t_{k+1} - t_k$ the *inter-event period*, and the interval (t_k, t_{k+1}) the *Additive Increase* (AI) phase.

Remark 3: The proposed scheduling scheme 1) prevents over-provisioning of available resources by limiting the workload share during the MD phase, and 2) meets the demand in finite time by permitting the admission of an ever-growing number of requests during the AI phase. These conflicting objectives, which are essential in typical scheduling and resource allocation systems, are attained by controlling the backlog of the central queue in an event-triggered fashion. The construction of meaningful cost functions guiding the selection of AIMD parameters for optimally achieving such objectives is possible [23].

Remark 4: The event-wise dynamics of the i^{th} admission controller can be written as

$$u_i(k+1) = \beta_i u_i(k) + \alpha_i T(k), \tag{6}$$

where $u_i(k) = u_i(t_k)$, $T(k) = t_{k+1} - t_k$, $0 \leq \beta_i < 1$, and $\alpha_i > 0$. We call the tuple (α_i, β_i) the AIMD parameters of the i^{th} node. Note that $T(k)$ is a function, among others, of the underlying workload. If $\sum_{i=1}^{n} \beta_i u_i(t_k) \geq \lambda(t_k)$, then, $T(k) = 0$, and $u_i(k+1) = \beta_i u_i(k)$. In this case, the additive increase phase of the controller is omitted and, instead, a sequence of multiplicative decreases occurs until requests are queued in the buffer.

Remark 5: The i^{th} control system (5)-(6) is event-triggered with varying period $T(k) \geq 0$. We assume that the admission rate function, $u_i(t)$, is entirely controlled by the i^{th} individual node and not by the aggregation node associated with the central queue. Under this assumption, a decentralised scheduling scheme is coordinated as follows. The term $\frac{1}{u_i(t)}$ specifies the interval between instants at which the i^{th} node calls for and admits a request. If a request, dispatched from the central queue upon i^{th} node's demand, reaches the i^{th} node, the latter stays in the AI phase, otherwise instantaneously enters the MD phase and, in parallel, calls for a new request. In other words, $u_i(t)$ ramps up with new request arrivals or drops after request call failures.

Remark 6: The AIMD-based algorithm proposed in the paper is in agreement with a synchronous setting [13]. In particular, we assume that all computing nodes go through the MD and AI phases simultaneously. An asynchronous setting can be modelled by allowing drop factors β_i to take on a set of different values [15]. Convergence properties of asynchronous schemes will be investigated in future work.

Remark 7: The classic AIMD algorithm [13] with capacity-related triggering conditions enjoys a decentralised setup at the expense of resource utilisation. This trade-off is due to the impossible communication between individual nodes and their lack of knowledge about the exact resource capacity. A similar trade-off is present in our AIMD scheme. The non-deterministic characteristics of the workload render the introduction of an extra queue necessary. Although this buffer adds memory to the system, potentially affecting the overall queueing overhead, it permits decentralised coordination and full workload dispatch.

By integrating $\dot{w}(t)$ in (1) in the interval $[t_k, t_{k+1}]$ with $\gamma_i(t)$ constant, and $u_i(t)$ given by (5), the event-wise dynamics of the local backlog is written as $w_i(k+1) = w_i(k) + (\beta_i u_i(k) + \frac{1}{2}\alpha_i T(k) - \gamma_i(k))T(k)$, where $\gamma_i(k)$ denotes the constant value of $\gamma_i(t)$ for $t \in [t_k, t_{k+1}]$. Letting $\alpha = (\alpha_1, \dots, \alpha_n)$, and $B = \mathrm{diag}(\beta_1, \dots, \beta_n)$, an event-driven discrete model of the system shown in Fig. 1, associated with the AIMD admission policy, is

$$\begin{bmatrix} u(k+1) \\ w(k+1) \end{bmatrix} = \begin{bmatrix} Bu(k) + \alpha T(k) \\ w(k) + (Bu(k) + \frac{\alpha}{2}T(k) - \gamma(k))T(k) \end{bmatrix}, \tag{7}$$

where $u(k)$ is the AIMD-controlled variable with triggering condition $\delta(t_k) = 0$, $T(k)$ is the inter-event period, and $\gamma(k)$ is the aggregate vector of service rates which are permitted to be altered at events. In Section V, we propose a stabilising feedback solution $\gamma(k)$ for decentralised resource allocation that ultimately allows model (7) to be simply tuned via the AIMD parameters. In Section IV, we focus on the admission control dynamics, which, as seen from (7), are decoupled from the aggregate vector $w(k)$.

IV. AIMD ADMISSION CONTROL

We consider a time-varying workload $\lambda(t) \in L = [\underline{\lambda},\ \overline{\lambda}] \subset \mathbb{R}_{>0}$, $t \geq 0$. We define the average workload between two instants, t_k and t_{k+1}, $t_{k+1} \geq t_k \geq 0$, as

$$\hat{\lambda}(k) = \begin{cases} \frac{1}{t_{k+1}-t_k}\int_{t_k}^{t_{k+1}} \lambda(t)\mathrm{d}t & \text{if } t_{k+1} > t_k, \\ \lambda(t_k) & \text{if } t_{k+1} = t_k. \end{cases} \tag{8}$$

Thus, we have $\hat{\lambda}(k) \in L = [\underline{\lambda},\ \overline{\lambda}]$.

Under the proposed triggering mechanism, an event occurs at t_k when the backlog is zero, $\delta(t_k) = 0$. By integrating $\dot{\delta}(t)$ given in (1) in the interval $[t_k, t_{k+1}]$ with boundary conditions $\delta(t_k) = \delta(t_{k+1}) = 0$, and with $T(k) = t_{k+1} - t_k$, and each $u_i(t)$ evolving according to (5), we have

$$\hat{\lambda}(k)T(k) - \sum_{i=1}^{n}(2\beta_i u_i(k) + \alpha_i T(k))\frac{T(k)}{2} = 0, \tag{9}$$

from which we derive that $T(k) = 2\frac{\hat{\lambda}(k) - \beta^\top u(k)}{1^\top \alpha}$, with $\hat{\lambda}(k)$ defined in (8). Since $\hat{\lambda}(k)$ can take any value in $[\underline{\lambda},\ \overline{\lambda}]$, we define the inter-event period as

$$T(k) = \max\left\{0,\ 2\frac{\hat{\lambda}(k) - \beta^\top u(k)}{1^\top \alpha}\right\}. \tag{10}$$

By substituting $T(k)$ in (6) for the right-hand side of (10), the aggregate admission control system is

$$u(k+1) = \begin{cases} Bu(k) & \text{if } \beta^\top u(k) \geq \hat{\lambda}(k) \\ \Phi u(k) + G\hat{\lambda}(k) & \text{if } \beta^\top u(k) < \hat{\lambda}(k), \end{cases} \tag{11}$$

where $u(0) \in \mathbb{R}^n_{>0}$, $\hat{\lambda}(k) \in L \subset \mathbb{R}_{>0}$, and

$$\Phi = B - \left(\frac{2}{1^\top \alpha}\right)\alpha\beta^\top, \tag{12}$$

$$G = \left(\frac{2}{1^\top \alpha}\right)\alpha, \tag{13}$$

with $\alpha \in \mathbb{R}^n$, $\beta \in \mathbb{R}^n$, $\alpha_i > 0$, $0 \leq \beta_i < 1$, $i = 1, \dots, n$, and $B \in \mathbb{R}^{n \times n}$ as defined in (7).

It is worth noting that the system (11) cannot exhibit a Zeno behaviour by the boundedness of the workload and the fact that its mode-1, which corresponds to $T(k) = 0$, is applied at most N successive times, with $N \leq \left\lceil \frac{\log \frac{\underline{\lambda}}{2\overline{\lambda}}}{\log \beta_{\max}} \right\rceil$ (see Proposition 4.2).

Remark 8: System (11) belongs to the family of piecewise affine systems, which are well-known for their inherent complexity [32]. However, this modelling approach permits a positive system representation, an essential property for an admission control scheme where individual controlled variables, $u_i(t)$, are non-negative quantities. Positivity of (11) is derived from (6) and (10). It is also worth noting that the dynamics (11) is continuous[2].

A. Convergence

Before showing that (11) has a uniformly bounded solution, $u(k)$, for all initial conditions $u(0) \in \mathbb{R}^n_{>0}$, we recall the following results.

Theorem 4.1 ([20, Theorem 1], [21, Section 5]): Let $C = D + \mu\zeta\zeta^\top$, where $D \in \mathbb{R}^{n \times n}$ is diagonal, $\mu \in \mathbb{R}$, and $\zeta \in \mathbb{R}^n$. Let $d_1 \leq d_2 \leq \dots \leq d_n$ be the eigenvalues of D, and $c_1 \leq c_2 \leq \dots \leq c_n$ be the eigenvalues of C. Then,

1) $d_1 \leq c_1 \leq d_2 \leq c_2 \leq \dots \leq d_n \leq c_n$ if $\mu > 0$,
2) $c_1 \leq d_1 \leq c_2 \leq d_2 \leq \dots \leq c_n \leq d_n$ if $\mu < 0$.

Theorem 4.1 underpins the development of the following result.

Theorem 4.2: Matrix Φ defined in (12) is Schur for all $\alpha_i > 0$, $0 \leq \beta_i < 1$, $i = 1, \dots, n$.

Proof: Stability of matrix Φ has be shown in [27] for $0 < \beta_i < 1$, $i = 1, \dots, n$. In Appendix I-A, we provide a complete proof allowing $\beta_i = 0$ for some $i = 1, \dots, n$. If $\beta_i = 0$ for all $i = 1, \dots, n$, the proof is trivial as $\Phi = 0$. ■

We are now in the position to state a stability result associated with system (11).

Theorem 4.3: Consider the matrix set $\Sigma = \{\Phi, B\}$, where Φ is defined in (12). Then, $\rho(\Sigma) < 1$, for all $0 \leq \beta_i < 1$, $\alpha_i \geq 0$, $i = 1, \dots, n$.

Proof: See Appendix I-B. ■

For simplicity, we write (11) as

$$u(k+1) = A_{\sigma(k)}u(k) + b_{\sigma(k)}\hat{\lambda}(k), \tag{14}$$

where $u(0) \in \mathbb{R}^n_{>0}$, the switching signal is defined as [3]

$$\sigma(k) = \begin{cases} 1 & \text{if } \beta^\top u(k) \geq \hat{\lambda}(k) \\ 2 & \text{if } \beta^\top u(k) < \hat{\lambda}(k), \end{cases} \tag{15}$$

and $A_1 = B$, $A_2 = \Phi$, $b_1 = 0 \in \mathbb{R}^n$, and $b_2 = G$.

Theorem 4.3 implies that the linear part of (14), namely, $u(k+1) = A_{\sigma(k)}u(k)$ is asymptotically stable for all $u(0) \in \mathbb{R}^n_{>0}$ and any switching sequence $\{\sigma(k)\}_0^\infty$.[4] This leads to the following corollary.

[2]Continuity is necessary for the construction of compact reachable sets.

[3]We abuse notation of $\sigma(\cdot)$ for brevity, as the switching signal is a function of the state $u(k)$ and $\hat{\lambda}(k)$, i.e., $\sigma(\cdot) : \mathbb{R}^{n+1}_{>0} \to \{1, 2\}$.

[4]This is sufficient, yet not necessary, to guarantee asymptotic stability of the linear part under constrained switching sequences. Each switching sequence is uniquely defined for a given initial condition $u(0)$ and an admissible sequence $\{\hat{\lambda}(k)\}_0^\infty$.

Corollary 3.1: The solution to (11), equivalently to (14), is uniformly bounded for all initial conditions $u(0) \in \mathbb{R}^n_{>0}$.

Proof: We show that the quantity $\lim_{k\to\infty} \|u(k)\|$ is bounded for any norm $\|\cdot\|$. For any $N \in \mathbb{N}$, $u(0) \in \mathbb{R}^n_{>0}$ and an admissible sequence $\hat{\lambda}(0), \hat{\lambda}(1), \ldots, \hat{\lambda}(N-1)$, we have

$$u(N) = \prod_{j=0}^{N-1} A_{\sigma(N-1-j)} u(0) + \sum_{j=0}^{N-1} \prod_{\nu=0}^{N-2-j} A_{\sigma(N-1-\nu)} b_{\sigma(j)} \lambda(j). \quad (16)$$

By Theorem 4.3, $\rho = \rho(A_1, A_2) < 1$, thus, for any norm $\|\cdot\|$ and all vectors $x \in \mathbb{R}^n$, there exist scalars $\Gamma \geq 1$ and $r > 0$ such that $\rho \leq r < 1$ and

$$\|\prod_{j=0}^{t-1} A_{\sigma(t-1-j)} x\| \leq \Gamma r^t \|x\|. \quad (17)$$

Consequently,

$$\begin{aligned}\|u(N)\| &\leq \Gamma r^N \|u(0)\| + \Gamma(1 + r + \ldots + r^{N-1}) \max_{\hat{\lambda}\in L} \|G\hat{\lambda}\| \\ &\leq \Gamma r^N \|u(0)\| + \frac{\Gamma}{1-r} \|G\overline{\lambda}\|. \end{aligned} \quad (18)$$

It follows that $\lim_{N\to\infty} \|u(N)\| \leq \frac{\Gamma\overline{\lambda}}{1-r}\|G\|$, which completes the proof. ∎

Ignoring switching constraints, the uniform boundedness of $u(k)$ is shown in Corollary 3.1, which results from Theorem 4.3. Taking account of the state-dependent constraints on the switching signal, a tighter bound on the solution $u(k)$ can be established. To this purpose, we introduce the following state augmentation. Let $z(k) = [u(k)^\top \ \hat{\lambda}(k)]^\top \in \mathbb{R}^{n+1}_{>0}$, and define matrices $\hat{\Phi} = \begin{bmatrix} \Phi & G \\ 0_{1\times n} & 0 \end{bmatrix}$, $\hat{B} = \begin{bmatrix} B & 0_{n\times 1} \\ 0_{1\times n} & 0 \end{bmatrix}$, $\hat{G} = \begin{bmatrix} 0_{n\times 1} \\ 1 \end{bmatrix}$. Then, a state-dependent switching system equivalent to (11) and (14) is written as

$$z(k+1) = F(z(k)), \quad (19)$$

where the map $F : \mathbb{R}^{n+1}_{>0} \to \mathbb{R}^{n+1}_{>0}$ is defined as

$$F(z(k)) = \begin{cases} \hat{B}z(k) + \hat{G}\eta(k), & \text{if } c^\top z(k) \geq 0, \\ \hat{\Phi}z(k) + \hat{G}\eta(k), & \text{if } c^\top z(k) < 0, \end{cases} \quad (20)$$

with $\eta(k) \in L \subset \mathbb{R}_{>0}$, $\hat{\lambda}(k+1) = \eta(k)$, and $c = \begin{bmatrix}\beta^\top & -1\end{bmatrix}^\top$. Asymptotic stability of the linear part of (19)-(20) follows from the following proposition.

Proposition 4.1: Consider matrices $\hat{\Phi}$, $\hat{B}$, as defined in (20). Then, the joint spectral radius of $\hat{\Sigma} = \{\hat{\Phi}, \hat{B}\}$ is $\rho(\hat{\Sigma}) < 1$.

Proof: Since $\hat{\Phi}$, $\hat{B}$ have a block upper-triangular form, from [22, Proposition 1.5], we have that $\rho(\hat{\Sigma}) = \max\{\rho(\{\Phi, B\}), \rho(0)\}$, i.e., $\rho(\hat{\Sigma}) = \rho(\{\Phi, B\})$. Then, by Theorem 4.3, $\rho(\{\hat{\Phi}, \hat{B}\}) < 1$. ∎

Next, we show that the system (19)-(20) possesses an invariant set reachable for any initial condition in finite time. We use the same notation for sets, i.e., $F(S) = \{F(x) : x \in S\}$, $S \subset \mathbb{R}^{n+1}$. We define

$$C_1 = \{z \in \mathbb{R}^{n+1} : z \geq 0, \ c^\top z \geq 0, \ z_{n+1} \in L\}, \quad (21)$$

$$C_2 = \{z \in \mathbb{R}^{n+1} : z \geq 0, \ c^\top z \leq 0, \ z_{n+1} \in L\}. \quad (22)$$

Proposition 4.2: With the notation introduced in (19) and (22), consider the set sequence

$$Z_{j+1} = Z_j \cup F(Z_j), \ Z_0 = C_2. \quad (23)$$

There exists $N \in \mathbb{N}$ such that Z_N is invariant with respect to (19)-(20). Also, the minimum value of N is upper-bounded by $\left\lceil \frac{\log \frac{\underline{\lambda}}{2\overline{\lambda}}}{\log \beta_{\max}} \right\rceil$, where $\underline{\lambda} \in \mathbb{R}_{>0}$ and $\overline{\lambda} \in \mathbb{R}_{>0}$ are the lower and upper bounds of $L \subset \mathbb{R}_{>0}$, respectively, and $\beta_{\max} = \max_{i\in\{1,\ldots,n\}} \beta_i$.

Proof: We denote by $F^M(X)$ the composition $F \circ F \circ \ldots \circ F$ of length M. First, we show that C_2 is always reachable from any initial condition in finite time. For any $z(0) = z_0 \in \mathbb{R}^n_{>0} \times L$ with $\|z_0\|$ finite, there exists $k^* \in \mathbb{N}$, which depends on $z_0 \in C_1$, such that $z(k^*) = F^{k^*}(z_0) \in C_2$. Suppose that this is not always true. Then, $\underline{\lambda} \leq z_{n+1}(k) \leq \sum_{i=1}^n \beta_i^k z_i(0)$ for all $k \geq 0$. Setting $\beta_{\max} = \max_{i=\{1,\ldots,n\}} \beta_i$, and $\bar{z} = \max_{i=\{1,\ldots,n\}} z_i(0)$, we may write $n\beta_{\max}^k \bar{z} \geq \underline{\lambda}$ or $k \log \beta_{\max} \geq \log \frac{\underline{\lambda}}{n\bar{z}}$ or $k \leq \frac{\log \frac{\underline{\lambda}}{n\bar{z}}}{\log \beta_{\max}}$ since $\log \beta_{\max} < 0$. However, the latter cannot always be true, since $k \geq 0$ can be arbitrarily large. Consider $z(0) \in C_2$. There is $N \in \mathbb{N}$, $j \in [1, N]$, such that $F^j(z(0)) \in C_2$. Since (23) can be written as $Z_i = \cup_{j=0}^i F^j(Z_0)$ with $Z_0 = C_2$, we have

$$Z_N = Z_0 \cup F(Z_0) \cup \ldots \cup F^{N-1}(Z_0). \quad (24)$$

If $z(0) \in Z_N \setminus F^{N-1}(Z_0)$, $z(1) \in Z_N$. If $z(0) \in F^{N-1}(Z_0)$, for any $\zeta \in F^{N-1}(Z_0)$ there is $y \in Z_0$ such that $\zeta = F^{N-1}(y)$. Then, for any $y \in Z_0$ there is $1 \leq i^* \leq N$ such that $F^{i^*}(y) \in Z_0$. Thus, we may write $F(\zeta) = F(F^{N-1}(y)) = F^N(y) = F^{N-i^*}(F^{i^*}(y)) = F^{N-i^*}(\xi)$ with $\xi \in Z_0$. Clearly $F^{N-i^*}(\xi) \in Z_N$.

To obtain an upper bound on the minimum value of N, we shall find the trajectory, originating at a point in C_2, that revisits C_2 after a course of multiplicative decreases with maximum length k^*. From (19)-(20),

$$\sum_{i=1}^n z_i(k) \leq 2\hat{\lambda}(k) \leq 2\overline{\lambda} \quad (25)$$

for all $k \geq 0$ and $z(0) \in C_2$. Consider $z(0) = z_0 \in C_2$ and $z(1) = F(z_0) \in C_1$, and let $\beta_{\max} = \max_{i\in\{1,\ldots,n\}} \beta_i$. Then, we require $\sum_{i=1}^n \beta_i^{k^*} z_i(1) \leq \hat{\lambda}(k^*)$, which is true if $\beta_{\max}^{k^*} \sum_{i=1}^n z_i(1) \leq \hat{\lambda}(k^*)$ or, from (25), if $\beta_{\max}^{k^*} 2\overline{\lambda} \leq \hat{\lambda}(k^*)$. The latter yields a maximum k^* if we require $\hat{\lambda}(k) = \underline{\lambda}$ for all $k \geq 1$, i.e., $\beta_{\max}^{k^*} 2\overline{\lambda} < \underline{\lambda}$ or $k^* > \frac{\log \frac{\underline{\lambda}}{2\overline{\lambda}}}{\log \beta_{\max}}$. Thus, the minimum value of N is less than or equal to $\left\lceil \frac{\log \frac{\underline{\lambda}}{2\overline{\lambda}}}{\log \beta_{\max}} \right\rceil$. ∎

We are now in the position to define the fixed point of (19)-(20) in the following theorem.

Theorem 4.4: With the notation introduced in (19), consider the set sequence $\{Y_j\}_j$ with

$$Y_0 = Z_N, \tag{26}$$
$$Y_{j+1} = F(Y_j), \tag{27}$$

where Z_N is constructed as in Proposition 4.2, and let $Y_\infty = \lim_{j\to\infty} Y_j$. Then, $Y_\infty = \bigcap_{j=0}^{\infty} Y_j$ exists.

Proof: The sets Y_j, $j \geq 0$, are compact; indeed, closedness follows by continuity of (19)-(20) on the switching surfaces and boundedness follows from the compactness of Y_0 and the piecewise affine dynamics. Moreover, the inclusions $Y_{j+1} \subseteq Y_j$, $j \geq 0$, hold by invariance of Y_0 with respect to (19)-(20) as shown in Proposition 4.2. Consequently, the sequence $\{Y_j\}_j$ is a nonincreasing sequence of compacts sets, and thus, has a limit Y_∞, with $Y_\infty = \bigcap_{j=0}^{\infty} Y_j$ [33, Lemma 1.8.2]. ■

A corollary of Theorem 4.4 follows.

Corollary 4.1: Suppose that there exists $k^* \in \mathbb{N}$ such that $Y_{k^*} \subseteq C_2$, where C_2 is defined in (22). Then, $Y_\infty \subseteq C_2$ is the minimal invariant set of the second mode of (19)-(20). Moreover, If C_2 is such that L is a singleton, i.e., $L = \{\lambda\}$, $\lambda \in \mathbb{R}_{>0}$, then

$$Y_\infty = \left[\left((I_n - \Phi)^{-1} G \lambda \right)^\top \ \lambda \right]^\top . \tag{28}$$

Proof: Since $Y_\infty \subseteq Y_k$ for all $k \geq 0$, by assumption, we have $Y_\infty \subseteq Y_{k^*} \subseteq C_2$. Also, the set sequence (26)-(27) for $j \geq k^*$ is obtained by

$$\hat{Y}_{\nu+1} = \hat{\Phi} \hat{Y}_\nu \oplus \hat{G} \lambda, \tag{29}$$

where $\nu \geq 0$, $\hat{Y}_0 = Y_{k^*}$. By linearity, and asymptotic stability of $\hat{\Phi}$, it follows that the limit set of (29), namely, $\hat{Y}_\infty$, exists [34] and is the minimal invariant set of the second mode of (19)-(20). If, also, L is deterministic and singleton, the sequence (29), originating at any point in C_2, converges to $Y_\infty = (I_{n+1} - \hat{\Phi})^{-1} \hat{G} \lambda$. Then, (28) is immediately derived in view of

$$(I_{n+1} - \hat{\Phi})^{-1} \hat{G} \lambda = \begin{bmatrix} (I_n - \Phi)^{-1} & (I_n - \Phi)^{-1} G \\ 0_{1\times n} & 1 \end{bmatrix} \begin{bmatrix} 0_{n\times 1} \\ 1 \end{bmatrix} \lambda.$$

■

B. Convexification of Y_∞

The construction of $\{Y_j\}_j$ is challenging as it is generically a sequence of non-convex sets. Calculating the sequence (26)-(27) recursively may be difficult to accomplish efficiently mainly due to the piecewise affine structure of the map F and the non-convexity of individual members, Y_j, $j \geq 1$. To this end, we present a convexified version of the sequence $\{Y_j\}_j$, the validity of which is stated next.

Proposition 4.3: Consider the set sequence

$$P^1_{j+1} = \mathrm{co}\left(F(P_j) \cap C_1 \right), \tag{30}$$
$$P^2_{j+1} = \mathrm{co}\left(F(P_j) \cap C_2 \right), \tag{31}$$
$$P_{j+1} = P^1_{j+1} \cup P^2_{j+1}, \tag{32}$$

with $P_0 = P^1_0 \cup P^2_0$, $P^1_0 = \mathrm{co}(Y_0 \cap C_1)$ and $P^2_0 = \mathrm{co}(Y_0 \cap C_2)$. Then,

$$P_j = \mathrm{co}(Y_j \cap C_1) \cup \mathrm{co}(Y_j \cap C_2), \tag{33}$$

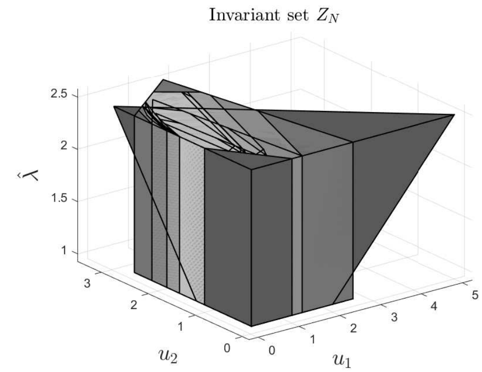

Fig. 2. Construction of Z_N, with $N = 6$, for AIMD system with $\alpha = (1, 2)$, $\beta = (0.5, 0.85)$, and $\hat{\lambda} \in [1, 2.5]$.

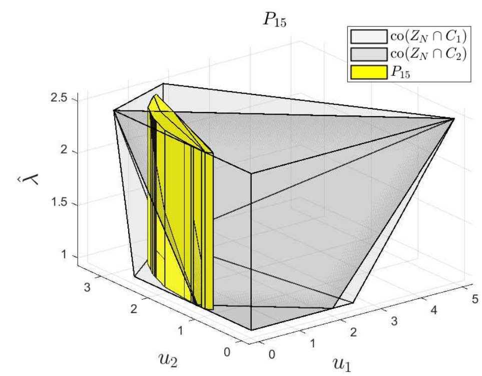

Fig. 3. Calculation of P_{15} of the sequence (30)-(32), with Z_N illustrated in Fig. 2, for AIMD parameters $\alpha = (1, 2)$, $\beta = (0.5, 0.85)$, and $\hat{\lambda} \in [1, 2.5]$.

for all $j \geq 0$.

Proof: See Appendix I-C. ■

Remark 9: Each set P_j is the union of two convex sets. Each of these two sets has a convex intersection with either C_1 or C_2. Thus, the sequence (30)-(32), originating at $P_0 = \mathrm{co}(Z_N \cap C_1) \cup \mathrm{co}(Z_N \cap C_2)$, can efficiently be calculated on standard computational geometry software. The set Z_N defined in (24), is the union of N non-convex sets with the ν^{th} member being $F^{\nu-1}(Z_0)$, $\nu = 1, \ldots, N$, consisting of one convex polyhedron if $\nu = \{1, 2\}$, or $2^{\nu-2}$, generically non-convex, polyhedra, otherwise.

Example 1: Consider a two-node system with AIMD parameters $\alpha = (1, 2)$, $\beta = (0.5, 0.85)$, and workload $\hat{\lambda} \in [1, 2.5]$. We wish to approximate Y_∞ via the sequence (30)-(32). We first construct the set Z_N, with $N = 6$, which is the union of 32 polyhedra as shown in Fig. 2. For $k_{\max} = 15$ iterations, we obtain P_{15} which is shown in Fig 3 in yellow. We also illustrate the orthogonal projection $U = \{u \in \mathbb{R}^2 : \exists \lambda \in L \text{ such that } [u^\top \ \lambda]^\top \in P_{15}\}$, in Fig. 4. Note that $P_{15} = U \times L$ by the structure of (19)-(20). Trajectories $u(k) \in \mathbb{R}^2$ for various admissible sequences $\{\hat{\lambda}(k)\}$ are also depicted in Fig. 4.

C. Constant workload

We now focus on the convergence properties of the trajectories of the system (11), under a constant workload. We first

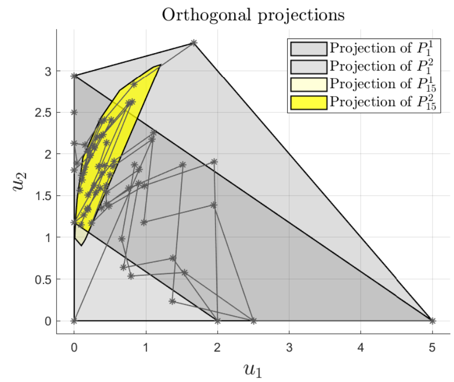

Fig. 4. Orthogonal projections of $\boldsymbol{P_1} = \boldsymbol{P_1^2} \cup \boldsymbol{P_1^2}$ and $\boldsymbol{P_{15}} = \boldsymbol{P_{15}^2} \cup \boldsymbol{P_{15}^2}$ with $\boldsymbol{P_1}$, $\boldsymbol{P_{15}}$ illustrated in Fig. 3, for AIMD parameters $\boldsymbol{\alpha} = \boldsymbol{(1, 2)}$, $\boldsymbol{\beta} = \boldsymbol{(0.5, 0.85)}$, and $\hat{\boldsymbol{\lambda}} \in \boldsymbol{[1, 2.5]}$.

prove the following invariance property with respect to the second mode of (11).

Proposition 4.4: Let

$$u^+ = \Phi u + G\lambda, \tag{34}$$

where u^+ denotes the successor state of u, the matrices $\Phi \in \mathbb{R}^{n\times n}$, $G \in \mathbb{R}^n$ are as defined in system (11), and $\lambda \in \mathbb{R}_{>0}$ is a positive constant. If

$$0 \le \beta_1 = \cdots = \beta_n < 1, \text{ or,} \tag{A1}$$

$$\sum_{i=1}^{n} \beta_i \alpha_i \le 0.5 \sum_{i=1}^{n} \alpha_i, \tag{A2}$$

with $\alpha_i > 0$, $0 \le \beta_i < 1$, $i = 1, \dots, n$, being the AIMD parameters, then, the set

$$U = \{u \in \mathbb{R}^n : \beta^\top u \le \lambda,\ 1^\top u \ge \lambda,\ u \ge 0\}, \tag{35}$$

is invariant with respect to (34).

Proof: If $u \in U$ then $1^\top u^+ \ge \lambda$ and $u^+ \ge 0$. To show that $\beta^\top u^+ \le \lambda$ when the AIMD parameters satisfy (A1) or (A2) we proceed as follows. Consider the linear program

$$\max_{u\in U} J(u), \text{ with } J(u) = \beta^\top(\Phi u + G\lambda). \tag{36}$$

By linearity of $J(u)$ in $u \in U$ and the convexity of U, the solution of (36) lies at the boundary of U. We can write

$$J(u) = \sum_{i=1}^{n} \beta_i u_i \left(\beta_i - 2\sum_{i=1}^{n} \beta_i \bar{\alpha}_i\right) + 2\sum_{i=1}^{n} \beta_i \bar{\alpha}_i \lambda, \tag{37}$$

where $\bar{\alpha}_i = \alpha_i/(1^\top \alpha)$. If (A1) is true, let $\bar{\beta} = \beta_1 = \beta_2 = \cdots = \beta_n$. Then, $J(u) = -\bar{\beta}^2 1^\top u + 2\bar{\beta}\lambda$ is maximised at any point $\bar{u} \in U$ for which $1^\top \bar{u} = \lambda$. Then, $J(\bar{u}) = \bar{\beta}(2-\bar{\beta})\lambda < \lambda$ since $\bar{\beta}(2-\bar{\beta}) < 1$ for all $0 \le \bar{\beta} < 1$. If (A2) is true, $\sum_{i=1}^n \beta_i\alpha_i \le 0.5\sum_{i=1}^n \alpha_i$, i.e., $2\sum_{i=1}^n \beta_i\bar{\alpha}_i \le 1$, we may write $J(u) = \sum_{i=1}^n \beta_i^2 u_i + 2\sum_{i=1}^n \beta_i\bar{\alpha}_i(\lambda - \sum_{i=1}^n \beta_i u_i) < \bar{J}(u) = \sum_{i=1}^n \beta_i^2 u_i + \lambda - \sum_{i=1}^n \beta_i u_i$ which is maximised at one of the n vertices $u^i = e_i\lambda$, where $e_i \in \mathbb{R}^n$ is the i^{th} vector of the canonical basis of $\mathbb{R}^n$. Clearly, $J(u) \le \bar{J}(u^i) = (\beta_i^2 - \beta_i + 1)\lambda \le \lambda$, which is true since $-.25 \le \beta_i^2 - \beta_i \le 0$ for all $0 \le \beta_i < 1$ ■

An immediate consequence of Proposition 4.4 is the convergence of all trajectories of the system (11) with constant workload to a unique fixed point. This is stated next.

Theorem 4.5: Let the AIMD system (11), equivalently (14), have a constant workload $\lambda^* \in \mathbb{R}_{>0}$. If (A1) or (A2) is true, then, for any initial condition $u(0) \in \mathbb{R}^n_{\ge 0}$, the underlying state trajectories converge to $u^* = (I - \Phi)^{-1}G\lambda^*$.

Proof: From Proposition 4.4, U defined in (35) is invariant with respect to $u(k+1) = \Phi u(k) + G\lambda^*$, and hence, if $u(0) \in U$, $\lim_{k\to\infty} u(k) = u^*$. Therefore, it suffices to show that all trajectories originating at $u(0) \notin U$ converge to U in finite time. Let $U_1 = \{u \in \mathbb{R}^n_{\ge 0} : \beta^\top u > \lambda,\ u \ge 0\}$, $U_2 = \{u \in \mathbb{R}^n_{\ge 0} : 1^\top u < \lambda,\ u \ge 0\}$, and $\bar{U} = U_1 \cup U_2$. Clearly, $U \cup \bar{U} = \mathbb{R}^n_{\ge 0}$. Consider that $u(0) \in U_1$ with finite $\|u(0)\|$. Similarly to Proposition 4.2, there is a finite $\kappa > 0$ such that $\beta^\top u(\kappa) \le \lambda$, with $u_i(\kappa) = \beta_i^\kappa u_i(0)$. Suppose $u(\kappa) \notin U$ but $u(\kappa) \in U_2$. Then, $u(\kappa + 1) = \Phi u(\kappa) + G\lambda^*$, and, by Proposition 4.4, $1^\top u(\kappa+1) > \lambda$ and $\beta^\top u(\kappa+1) < \lambda$, i.e., $u(\kappa + 1) \in U$. Thus, for any initial condition, the system trajectories enter U in finite time and converge to $u^* = (I - \Phi)^{-1}G\lambda^*$. The uniqueness of u^* is a result of the invertibility of $I - \Phi$ following from $\rho(\Phi) < 1$ by Theorem 4.2. ■

We claim that Theorem 4.5 is true under much milder assumptions. In fact, from extensive simulations using different system dimensions and AIMD-parameters configurations, the set sequence (26)-(27) always converges to a unique fixed point when the workload is constant, if $\alpha_i > 0$, $0 \le \beta_i < 1$, $i = 1, \dots, n$. This is stated formally next.

Conjecture 4.1: Consider the AIMD system (11) with a constant workload $\hat{\lambda}(k) = \lambda \in \mathbb{R}_{>0}$ for all $k \ge 0$. Then, for any value of the parameters $\alpha_i > 0$, $0 \le \beta_i < 1$, $i = 1, \dots, n$, the solution (16) converges to a unique fixed point $u^* = (I - \Phi)^{-1}G\lambda$.

Remark 10: The fixed point claimed in Conjecture 4.1 is written element-wise as $u_i^* = \frac{\alpha_i}{1-\beta_i}T^*$, where $T^* > 0$ is the associated inter-event period. Under Conjecture 4.1, or when conditions (A1) or (A2) hold, the fixed point u^* can be controlled by appropriately tuning the AIMD parameters (α_i, β_i), $i = 1, \dots, n$, permitting objectives, such as *weighted fairness*, to be efficiently achieved.

Example 2: We consider an AIMD system with 1000 nodes and constant workload $\lambda = 1$. Let individual AIMD parameters be randomly drawn from the uniform distribution in the intervals $0.1 \le \alpha_i \le 10$ and $0 \le \beta_i \le 0.9$, respectively. Convergence and the randomly picked weighted fairness are demonstrated in Fig. 5.

V. Resource Allocation Control

Resource allocation in DCNs pertains to strategies ensuring that critical performance metrics (e.g., execution and response times) are bounded as more requests are added to the system. In this section, we provide a resource allocation law for controlling individual service rates $\gamma_i(k)$, $i = 1, \dots, n$. We focus on stability as a qualitative property, in the absence of which, other objectives may be impossible to attain. We present a resource allocation strategy coupled with the admission control

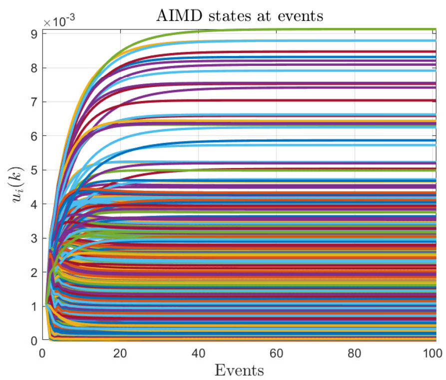

Fig. 5. Convergence and weighted fairness of the one-thousand-node AIMD system described in Example 2.

policy (11), originally introduced for constant workload in [27]. We show that the proposed resource allocation control law is compatible with time-varying workload profiles.

First, we provide an upper bound on the inter-event period $T(k)$. We rewrite (10) as

$$T(k) = \max\left\{0,\ 2\frac{-c^\top z(k)}{1^\top a}\right\}, \tag{38}$$

where c is as defined in (20), and $z(k)$ is the solution to (19)-(20) with $z(0) \in C_1 \cup C_2$. From Proposition 4.2, for any initial condition $z(0)$, the solution $z(k)$ is confined in Z_N, defined in (24), which is invariant with respect to (19)-(20). Thus, we focus on an upper bound on $T(k)$ obtained by maximising (38) over $z(k) \in Z_N$. From the invariance and compactness of Z_N, we derive boundedness of $T(k)$ being a piecewise linear function of $z(k)$. Also, $Z_N = (Z_N \cap C_1) \cup (Z_N \cap C_2)$. Clearly, if $z(k) \in Z_N \cap C_1$, $T(k) = 0$. Thus, $T(k)$ is linear in $z(k)$ for $z(k) \in Z_N \cap C_2$ and an upper bound may be obtained efficiently. This is provided by the following linear program with convex constraints:

$$\max_z \left\{2\frac{-c^\top z}{1^\top \alpha}\right\} \text{ subject to } z \in \mathrm{co}(Z_N \cap C_2). \tag{39}$$

We denote by $T_{\max}$ the upper bound obtained from (39). We are now in a position to state the following theorem.

Theorem 5.1: Consider a workload $\hat{\lambda}(k) \in L = [\underline{\lambda},\ \overline{\lambda}] \subset \mathbb{R}_{>0}$ for all $k \geq 0$, and let (α_i, β_i) be the AIMD parameters, with $\alpha_i > 0$, $0 \leq \beta_i < 1$, $i = 1, \ldots, n$, and $\gamma_i(k)$, $i = 1, \ldots, n$, denote service rates at the k^{th} event, respectively. The event-based dynamics of the backlog of the i^{th} node is written as

$$w_i(k+1) = w_i(k) + (\beta_i u_i(k) + \frac{\alpha_i}{2}T(k) - \gamma_i(k))T(k), \tag{40}$$

where $w_i(0) \geq 0$, $u_i(k)$ is the i^{th} state of system (11), $T(k)$ is the associated inter-event period, and $k \geq k^*$ such that $[u(k)^\top\ \hat{\lambda}(k)]^\top \in Z_N$, and $T(k) \leq T_{\max}$. Let the resource allocation policy be

$$\gamma_i(k) = \beta_i u_i(k) + \sqrt{2\alpha_i w_i(k)}. \tag{41}$$

Then, the set $\mathcal{W}_i = [0,\ \frac{a_i}{2}T_{\max}^2]$ is invariant with respect to (40)-(41). Also, for $w_i(0) \notin \mathcal{W}_i$, there is $\kappa > 0$ such that $w_i(\kappa) \in \mathcal{W}_i$.

Proof: One may check that (40)-(41) is nonnegative, and $w_i(k+1) \in [0,\ \frac{\alpha_i}{2}T(k)^2]$ if $w_i(k) \in [0,\ \frac{\alpha_i}{8}T(k)^2]$ and $w_i(k+1) \leq w_i(k)$ if $w_i(k) \geq \frac{\alpha_i}{8}T(k)^2$. Since, $T_{\max} \geq T(k)$ for all $k \geq k^*$, we have that $w_i(k+1) \in \mathcal{W}_i$ if $w_i(k) \in [\frac{\alpha_i}{2}T(k)^2,\ \frac{\alpha_i}{2}T_{\max}^2]$, and hence, $\mathcal{W}_i$ is invariant. The last statement of the theorem follows from [27, Theorem 3]. ■

We highlight appealing characteristics of the proposed resource allocation scheme as follows.

Remark 11: The positivity of (40)-(41) implies that resources are not over-provisioned under the control (41). Trajectories of (40)-(41) are globally attracted to $\mathcal{W}_i = [0,\ \frac{a_i}{2}T_{\max}^2]$ in finite time. See Example 3. The bound on $T_{\max}$ may be improved by solving (39) over $z \in \mathrm{co}(Y_\infty \cap C_2)$. The boundedness of local backlogs and the boundedness of the AIMD system trajectories make it feasible to obtain bounds on performance metrics, e.g., queueing time. The latter may be computed locally, see [27, Section VI].

Remark 12: The resource allocation (41) is decentralised as only local information is required. Moreover, it is scalable with respect to the number of computing nodes. Stability properties of (40)-(41) are independent of the particular tuning of AIMD parameters α_i, β_i, and the workload function $\hat{\lambda}(k) \in L$.

Example 3: Consider a system with three nodes, AIMD parameters $\alpha = (1, 1.5, 2)$, $\beta = (0.8, 0.5, 0.3)$, varying workload $\hat{\lambda} \in [90, 100]$, and initial local backlog $w(0) = (10, 20, 30)$. Let individual backlogs be regulated via the resource allocation control (41). The evolution of local backlogs, $w_i(k)$, $i = 1, 2, 3$, $k \geq 0$, is depicted in Fig. 6. By Theorem 5.1, invariant sets are computed as $\mathcal{W}_1 = [0, 125.08]$, $\mathcal{W}_2 = [0, 187.63]$, $\mathcal{W}_3 = [0, 250.17]$, with $T_{\max} = 15.82$ obtained by (39). The maximum values of the sets $\mathcal{W}_i$, $i = 1, 2, 3$, are illustrated with dashed lines. Tighter bound on $T_{\max}$ is attained by solving (39) over an approximation of $\mathrm{co}(Y_\infty \cap C_2)$, as pointed out in Remark 11. The improved maximum values of $\mathcal{W}_i$, $i = 1, 2, 3$, are shown with solid lines.

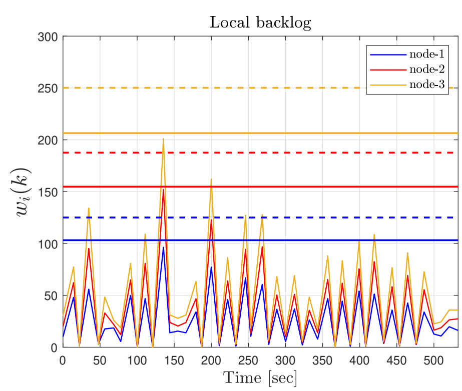

Fig. 6. Local backlogs of a three-node system described in Example 3.

VI. Comparative study

The proposed control scheme has been successfully integrated into Kubernetes[1] as the core architecture of a resource autoscaling mechanism established in a small edge infrastructure [10]. In addition to demonstrating the superiority of the proposed framework to Kubernetes' baseline solution, the experiments presented in [10] also highlight how the framework can be customised to address practical challenges such as workload identification and quantification of scheduling decision variables. Here, we provide a direct comparison of the proposed AIMD-based solution against standard scheduling principles, namely, the *Weighted Random* (WR) scheduler, the *Weighted Round-Robin* (WRR) scheduler [35], and the *Join-the-Shortest-Queue* (JSQ) scheduler [36]. Our numerical experiment is carried out in Matlab, hence we avoid noisy observations potentially emerging in real deployments. According to the WR scheduler, a request is assigned to a node randomly (e.g., by flipping a biased coin), whereas according to the JSQ scheduler, a request joins the node with the least backlog (in case of ties, WR scheduling is applied). The WRR scheduler has a cyclic behaviour where, in each cycle, a request joins a node based on the previous choice and the weights associated with the nodes.

In our experiment, we consider a Poisson workload profile with a mean inter-arrival rate λ [requests/sec], that is, a request flow with random inter-arrival intervals drawn from an exponential distribution with mean value $\frac{1}{\lambda}$. This stochastic workload setup differs from the analysis in the previous sections, highlighting the versatility of our approach. We consider a three-node system with node-2 having two times the resource size of node-1 and node-3 having three times the resource size of node-1. In our numerical study, whenever a non-AIMD solution is considered, the service rate of each node is fixed. Specifically, we choose $\gamma_1 = \frac{1}{6}\lambda\rho$, where $\rho > 1$, $\gamma_2 = 2\gamma_1$ and $\gamma_3 = 3\gamma_1$. Let $p(j)$ be the node index to which the jth request is assigned. According to the WRR scheduler, the node index is updated as $p(j) = \mod(j-1, 6)+1$, where $p(j) \in \{1\}$ indicates assignment to node-1, $p(j) \in \{2, 3\}$ indicates assignment to node-2, and $p(j) \in \{4, 5, 6\}$ indicates assignment to node-3.

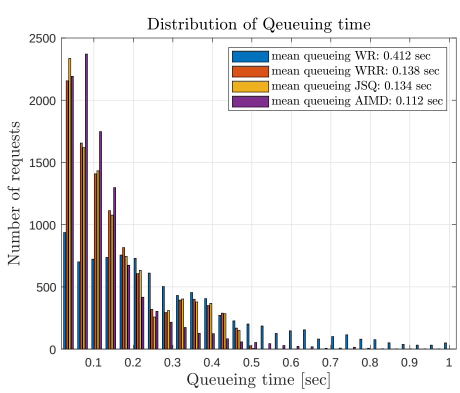

Fig. 7. Queueing times of $\mathbf{10000}$ requests entering a three-node system as a Poisson workload with a mean inter-arrival rate $\boldsymbol{\lambda} = \mathbf{100}$ [requests/sec], and scheduled according to the WR, WRR, JSQ, and AIMD methods.

We assess the performance of the four schemes by measuring the resulting queueing time at the request level for a Poisson workload with a mean inter-arrival rate $\lambda = 100$ [requests/sec] and a total duration of 100 seconds. In the WR, WRR, and JSQ solutions, the requests are assigned to nodes right after their arrivals. In contrast, in the AIMD system, the requests are required to wait for a period of time before being dispatched to a computing node, thereby inducing a queueing overhead. We report the results of an AIMD configuration against the non-AIMD schedulers in Fig. 7, where the distributions of queueing times are shown as histograms. In the figure, we denote by AIMD a three-node system with AIMD parameters $\alpha = (40, 80, 120)$, $\beta = (0.5, 0.5, 0.5)$.

By comparing the histograms of the four schemes in Fig. 7, the worst performance is observed for the WR scheduler, which indicates the highest mean queueing time (0.412 sec) and the widest dispersion. The WRR and JSQ schedulers result in similar performances producing mean queueing times of 0.138 sec and 0.134 sec, respectively, and similar dispersion. Guided by simulations, our choice of AIMD parameters results in an AIMD solution that outperforms the other three schemes achieving the lowest mean queueing time, namely, 0.112 sec, and significantly less average dispersion. Last, it is worth noting that the WR, WRR, and JSQ algorithms lead to overprovisioning of resources on average for 12%, 9%, and 7% of the time, respectively, whereas the AIMD method, by construction, prevents this phenomenon.

VII. Conclusion

We have proposed an AIMD-based control method for general scheduling and load-balancing problems in distributed computing networks. By introducing a central queue acting as the aggregation point of requests, we design an admission control system with AIMD dynamics that is triggered whenever the central queue vanishes. Under this triggering condition, requests are distributed in full, thus overcoming the limitation of a straightforward application of the AIMD algorithm. The resulting AIMD control scheme is a positive constrained switching system, which is convergent under peak bounded workload profiles irrespective of the AIMD parameters or the number of nodes. In view of the simplicity of the scheduling method, we have proposed a stabilising resource allocation strategy with decentralised architecture leading to a closed-loop scheduling and resource allocation control system configurable simply by the set of AIMD parameters. Establishing an optimisation framework for guiding the selection of AIMD parameters driven by meaningful trade-offs and cost functions related to, e.g., the maximum triggering frequency and the average queueing time, is important and will be pursued in future work.

References

[1] P. Mach and Z. Becvar, "Mobile Edge Computing: A Survey on Architecture and Computation Offloading," *IEEE Communications Surveys and Tutorials*, vol. 19, no. 3, pp. 1628–1656, 2017.

[2] M. Maggio, H. Hoffmann, M. D. Santambrogio, A. Agarwal, and A. Leva, "Controlling software applications via resource allocation within the Heartbeats framework," in *Proceedings of the 49th IEEE Conference on Decision and Control*, 2010, pp. 3736–3741.

[3] E. Kalyvianaki, T. Charalambous, and S. Hand, "Adaptive resource provisioning for virtualized servers using kalman filters," *ACM Transactions on Autonomous and Adaptive Systems*, vol. 9, no. 2, pp. 1–35, 2014.

[4] E. Makridis, K. Deliparaschos, E. Kalyvianaki, A. Zolotas, and T. Charalambous, "Robust Dynamic CPU Resource Provisioning in Virtualized Servers," *IEEE Transactions on Services Computing*, vol. 15, no. 2, pp. 956–969, 2022.

[5] Z. Wang, X. Zhu, and S. Singhal, "Utilization and SLO-Based Control for Dynamic Sizing of Resource Partitions," in *Ambient Networks*, J. Schönwälder and J. Serrat, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 133–144.

[6] M. Avgeris, D. Dechouniotis, N. Athanasopoulos, and S. Papavassiliou, "Adaptive resource allocation for computation offloading: A control-theoretic approach," *ACM Transactions on Internet Technology*, vol. 19, no. 2, pp. 1–20, 2019.

[7] J. Tang, W. P. Tay, and Y. Wen, "Dynamic request redirection and elastic service scaling in cloud-centric media networks," *IEEE Transactions on Multimedia*, vol. 16, no. 5, pp. 1434–1445, 2014.

[8] T. Sandholm and K. Lai, "Dynamic Proportional Share Scheduling in Hadoop," in *Job Scheduling Strategies for Parallel Processing*, E. Frachtenberg and U. Schwiegelshohn, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 110–131.

[9] S. Ranjan and E. Knightly, "High-performance resource allocation and request redirection algorithms for web clusters," *IEEE Transactions on Parallel and Distributed Systems*, vol. 19, no. 9, pp. 1186–1200, 2008.

[10] D. Spatharakis, I. Dimolitsas, E. Vlahakis, D. Dechouniotis, N. Athanasopoulos, and S. Papavassiliou, "Distributed Resource Autoscaling in Kubernetes Edge Clusters," in *18th International Conference on Network and Service Management*. IEEE, 2022, pp. 163–169.

[11] A. N. Tantawi and D. Towsley, "Optimal Static Load Balancing in Distributed Computer Systems," *Journal of the ACM*, vol. 32, no. 2, pp. 445–465, 1985.

[12] S. H. Low and D. E. Lapsley, "Optimization flow control - I: Basic algorithm and convergence," *IEEE/ACM Transactions on Networking*, vol. 7, no. 6, pp. 861–874, 1999.

[13] D. M. Chiu and R. Jain, "Analysis of the increase and decrease algorithms for congestion avoidance in computer networks," *Computer Networks and ISDN Systems*, vol. 17, no. 1, pp. 1–14, 1989.

[14] M. Corless, C. King, R. Shorten, and F. Wirth, *AIMD Dynamics and Distributed Resource Allocation*. Society for Industrial and Applied Mathematics, 2016.

[15] R. N. Shorten, D. J. Leith, J. Foy, and R. Kilduff, "Analysis and design of AIMD congestion control algorithms in communication networks," *Automatica*, vol. 41, no. 4, pp. 725–730, 2005.

[16] R. Shorten, F. Wirth, and D. Leith, "A positive systems model of TCP-like congestion control: Asymptotic results," *IEEE/ACM Transactions on Networking*, vol. 14, no. 3, pp. 616–629, 2006.

[17] S. Studli, E. Crisostomi, R. Middleton, and R. Shorten, "A flexible distributed framework for realising electric and plug-in hybrid vehicle charging policies," *International Journal of Control*, vol. 85, no. 8, pp. 1130–1145, 2012.

[18] M. Corless and R. Shorten, "An ergodic AIMD algorithm with application to high-speed networks," *International Journal of Control*, vol. 85, no. 6, pp. 746–764, 2012.

[19] A. Berman, R. Shorten, and D. Leith, "Positive matrices associated with synchronised communication networks," *Linear Algebra and Its Applications*, vol. 393, no. 1-3, pp. 47–54, 2004.

[20] J. R. Bunch, C. P. Nielsen, and D. C. Sorensen, "Rank-one modification of the symmetric eigenproblem," *Numerische Mathematik*, vol. 31, no. 1, pp. 31–48, 1978.

[21] G. H. Golub, "Some Modified Matrix Eigenvalue Problems," *SIAM Review*, vol. 15, no. 2, pp. 318–334, 1973.

[22] R. Jungers, *The Joint Spectral Radius*, ser. Lecture Notes in Control and Information Sciences. Springer Berlin Heidelberg, 2009, vol. 385.

[23] W. Ren, E. Vlahakis, N. Athanasopoulos, and R. Jungers, "Optimal Resource Scheduling and Allocation in Distributed Computing Systems," in *Proceedings of the 2022 American Control Conference*. IEEE, 2022, pp. 2327–2332.

[24] S. T. Maguluri and R. Srikant, "Scheduling jobs with unknown duration in clouds," in *Proceedings of the IEEE INFOCOM*, 2013, pp. 1887–1895.

[25] S. T. Maguluri, R. Srikant, and L. Ying, "Heavy traffic optimal resource allocation algorithms for cloud computing clusters," *Performance Evaluation*, vol. 81, pp. 20–39, 2014.

[26] F. Blanchini and S. Miani, *Set-Theoretic Methods in Control*, ser. Systems & Control: Foundations & Applications. Birkhäuser, 2015.

[27] E. Vlahakis, N. Athanasopoulos, and S. McLoone, "AIMD scheduling and resource allocation in distributed computing systems," in *Proceedings of the 60th IEEE Conference on Decision and Control*, 2021, pp. 4642–4647.

[28] L. Kleinrock, *Queueing systems, Volume I: Theory*. Wiley-Interscience, 1975.

[29] C. G. Cassandras and S. Lafortune, *Introduction to discrete event systems*. Springer US, 2008.

[30] W. Xin, J. Sun, G. Wang, and L. Dou, "A Mixed Switching Event-Triggered Transmission Scheme for Networked Control Systems," *IEEE Transactions on Control of Network Systems*, vol. 9, no. 1, pp. 390–402, 2022.

[31] X. Wang, J. Sun, G. Wang, F. Allgöwer, and J. Chen, "Data-Driven Control of Distributed Event-Triggered Network Systems," *IEEE/CAA Journal of Automatica Sinica*, vol. 10, no. 2, pp. 351–364, 2023.

[32] S. V. Raković, P. Grieder, M. Kvasnica, D. Q. Mayne, and M. Morari, "Computation of invariant sets for piecewise affine discrete time systems subject to bounded disturbances," in *Proceedings of the 43rd IEEE Conference on Decision and Control*, 2004, pp. 1418–1423.

[33] R. Schneider, *Convex Bodies: The Brunn–Minkowski Theory*, 2nd ed. Cambridge University Press, 2013.

[34] I. Kolmanovsky and E. G. Gilbert, "Theory and computation of disturbance invariance sets for discrete-time linear systems," *Mathematical Problems in Engineering*, vol. 4, pp. 317–367, 1998.

[35] W. Wang and G. Casale, "Evaluating Weighted Round Robin Load Balancing for Cloud Web Services," in *16th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing*. Timisoara, Romania: IEEE, 2014, pp. 393–400.

[36] V. Gupta, M. Harchol Balter, K. Sigman, and W. Whitt, "Analysis of join-the-shortest-queue routing for web server farms," *Performance Evaluation*, vol. 64, no. 9, pp. 1062–1081, 2007.

APPENDIX I

A. Proof of Theorem 4.2

Matrix Φ is Schur if $0 < \beta_i < 1$, $i = 1, \dots, n$, see [27, Theorem 2]. Without loss of generality, assume that $\beta_1 = \ldots = \beta_{p-1} = 0$ and $0 < \beta_i < 1$, $i = p, \dots, n$. Then, it is easy to see that Φ is a block upper-triangular matrix with diagonal blocks $\{0, \dots, 0, \Phi_p\}$ where $\Phi_p = \text{diag}(\beta_p, \dots, \beta_n) - \frac{2}{1^\top \alpha}(\alpha_p, \dots, \alpha_n)^\top(\beta_p, \dots, \beta_n)$, and $\sigma(\Phi) = \{0, \dots, 0, \sigma(\Phi_p)\}$. Thus, we only need to show that Φ_p is a Schur matrix. Note that $\Phi_p = (I - 2A_p)B_p$, where $A_p = \hat{\alpha}_p 1^\top$ with $\hat{\alpha}_p = (\alpha_p, \dots, \alpha_n)/1^\top\alpha$, and $B_p = \text{diag}(\beta_p, \dots, \beta_n)$. Note also that A_p is a rank-one matrix with $\sigma(A_p) = \{\hat{\alpha}_p^\top 1, 0, \dots, 0\}$ and $0 < \hat{\alpha}_p^\top 1 < 1$. Thus, $\sigma(I - 2A_p) = \{\xi, 1, \dots, 1\}$ with $\xi = 1 - 2\hat{\alpha}_p^\top 1$ and $-1 < \xi < 1$. We consider the following two cases.

i) Let $\xi = 0$. Then, $\det(\Phi_p) = 0$ since $\det(\Phi_p) = \det(I - 2A_p)\det(B_p)$ and $\sigma(I - 2A_p) = \{0, 1, \dots, 1\}$. This implies that the product $\phi_p\phi_{p+1}\ldots\phi_n = 0$, where $\{\phi_p, \dots, \phi_n\} = \sigma(\Phi_p)$. Letting now $\hat{A}_p = \text{diag}(\bar{\alpha}_p, \dots, \bar{\alpha}_n)$, where $\bar{\alpha}_i = \alpha_i/1^\top\alpha$, $\forall i$, and defining $\hat{T}_p = \hat{A}_p^{\frac{1}{2}} B_p^{-\frac{1}{2}}$, it is true that $\sigma(\Phi_p) = \sigma(\hat{\Phi}_p)$ where $\hat{\Phi}_p = \hat{T}_p^{-1}\Phi_p\hat{T}_p$ and $\hat{\Phi}_p = B_p - 2z_p z_p^\top$ with $z_p = (\sqrt{\bar{\alpha}_p\beta_p}, \dots, \sqrt{\bar{\alpha}_n\beta_n})$. Without loss of generality, assume that $\phi_p \le \phi_{p+1} \le \cdots \le \phi_n$. Then, by Theorem 4.1, we may write that

$$\phi_p \le \beta_p \le \phi_{p+1} \le \beta_{p+1} \le \ldots \le \phi_n \le \beta_n. \tag{42}$$

Thus, $0 < \phi_i < 1$, $i = p+1, \dots, n$ and $\phi_p = 0$, implying that Φ_p is a Schur matrix.

ii) Let now $\xi \neq 0$. Then, (42) still holds implying that $0 < \phi_i < 1$, $i = p+1, \dots, n$. Then, since $\det(\Phi_p) = \det(I - 2A_p)\det(B_p)$ we have that $\phi_p\phi_{p+1}\ldots\phi_n = \xi\beta_p\beta_{p+1}\ldots\beta_n$ which further implies that $|\phi_p| \le |\xi|\beta_n < 1$ since $\phi_{p+1}\ldots\phi_n \ge \beta_p\ldots\beta_{n-1}$ from (42). Thus, matrix Φ_p is Schur. This completes the proof.

B. Proof of Theorem 4.3

We will consider the following two cases. i) Assume that $0 < \beta_i < 1$ $\forall i$. Define matrix $A = \frac{1}{1^\top\alpha}\text{diag}(\alpha_1, \dots, \alpha_n)$ and transformation matrix $T = A^{\frac{1}{2}}B^{-\frac{1}{2}}$, and let $\bar{\Sigma} = T^{-1}\Sigma T = \{\bar{\Phi}, \bar{B}\}$, where $\bar{\Phi} = T^{-1}\Phi T$ and $\bar{B} = T^{-1}BT = B$. Due to invariance of the joint spectral radius under a similarity transformation [22, Proposition 1.3], we have $\rho(\Sigma) = \rho(\bar{\Sigma})$. Matrix $\bar{\Phi}$ is a symmetric matrix. Thus, $\bar{\Sigma}$ is a set of two symmetric matrices and, thus, from [22, Corollary 2.3], we have $\rho(\bar{\Sigma}) = \max\{\rho(\bar{\Phi}), \rho(B)\}$. Clearly, since $\bar{\Phi}, B$ are Schur matrices, it follows that $\rho(\bar{\Phi}) < 1$, $\rho(B) < 1$, and, thus, $\rho(\bar{\Sigma}) = \rho(\Sigma) < 1$. ii) Without loss of generality, assume that $\beta_1 = \beta_2 = \ldots = \beta_{p-1} = 0$ and $0 < \beta_i < 1$, $i = p, \ldots, n$. It is easy to see that $\Sigma = \{\Phi, B\}$ is a set of two block upper-triangular matrices with their lower diagonal blocks being of identical dimensions. In fact, $\Phi = \begin{bmatrix} 0_{(p-1)\times(p-1)} & * \\ 0_{(n-p+1)\times(n-p+1)} & \Phi_p \end{bmatrix}$ with $\Phi_p \in \mathbb{R}^{(n-p+1)\times(n-p+1)}$ as defined in the proof of Theorem 4.2, and $B = \operatorname{diag}(0_{(p-1)\times(p-1)}, B_p)$ with $B_p = \operatorname{diag}(\beta_p, \beta_{p+1}, \ldots, \beta_n) \in \mathbb{R}^{(n-p+1)\times(n-p+1)}$. From [22, Proposition 1.5], we have that $\rho(\Sigma) = \rho(\{\Phi_p, B_p\})$, while from [22, Proposition 1.3] we have that $\rho(\{\Phi_p, B_p\}) = \rho(\{\hat{\Phi}_p, \hat{B}_p\})$ where $\hat{\Phi}_p = T_p^{-1}\Phi_p T_p$ and $\hat{B}_p = T_p^{-1}B_p T_p = B_p$, with matrix T_p similarly defined as $\hat{T}_p$ in the proof of Theorem 4.2. Since $\hat{\Sigma}_p = \{\hat{\Phi}_p, B_p\}$ is a set of symmetric matrices, from [22, Corollary 2.3], we have $\rho(\hat{\Sigma}_p) = \max\{\rho(\Phi_p), \rho(B_p)\}$ for which, by Theorem 4.2, we know that $\rho(\hat{\Sigma}_p) < 1$. Thus, $\rho(\Sigma) = \rho(\hat{\Sigma}_p) < 1$.

C. Proof of Proposition 4.3

Considering (30) for $j = 0$, we get $P_1^1 = \operatorname{co}\big(F(\operatorname{co}(Y_0 \cap C_1) \cup \operatorname{co}(Y_0 \cap C_2)) \cap C_1\big) = \operatorname{co}\big(F\big(\operatorname{co}(Y_0 \cap C_1)\big) \cup F\big(\operatorname{co}(Y_0 \cap C_2)\big) \cap C_1\big)$. Since $F\big(\operatorname{co}(Y \cap C_1)\big) = \hat{B}\operatorname{co}(Y \cap C_1) \oplus \hat{G}L$ and $F\big(\operatorname{co}(Y \cap C_2)\big) = \hat{\Phi}\operatorname{co}(Y \cap C_1) \oplus \hat{G}L$ are affine maps, we have that $F\big(\operatorname{co}(Y_0 \cap C_1)\big) = \operatorname{co}\big(F(Y_0 \cap C_1)\big)$ and $F\big(\operatorname{co}(Y_0 \cap C_2)\big) = \operatorname{co}\big(F(Y_0 \cap C_2)\big)$. Thus,

$$P_1^1 = \operatorname{co}\big(\operatorname{co}\big(F(Y_0 \cap C_1)\big) \cup \operatorname{co}\big(F(Y_0 \cap C_2)\big) \cap C_1\big). \quad (43)$$

We may also write

$$\begin{aligned} &\operatorname{co}\big(F(Y_0) \cap C_1\big) = \operatorname{co}\big(F\big((Y_0 \cap C1) \cup (Y_0 \cap C_2)\big) \cap C_1\big) \\ &= \operatorname{co}\big(F(Y_0 \cap C1) \cup F(Y_0 \cap C_2) \cap C_1\big) \\ &= \operatorname{co}\big(\operatorname{co}\big(F(Y_0 \cap C1)\big) \cup \operatorname{co}\big(F(Y_0 \cap C_2)\big) \cap \operatorname{co}(C_1)\big) \\ &= \operatorname{co}\big(\operatorname{co}\big(F(Y_0 \cap C1)\big) \cup \operatorname{co}\big(F(Y_0 \cap C_2)\big) \cap C_1\big). \quad (44) \end{aligned}$$

From (43), (44), we have $P_1^1 = \operatorname{co}(F(Y_0) \cap C_1) = \operatorname{co}(Y_1 \cap C_1)$. Similarly, we obtain $P_1^2 = \operatorname{co}(Y_1 \cap C_2)$, which allows us to write $P_1 = \operatorname{co}(Y_1 \cap C_1) \cap \operatorname{co}(Y_1 \cap C_2)$. Following the same line of reasoning and applying (32) for all $j \geq 1$, the result follows.

Acknowledgment

The authors would like to thank the anonymous reviewers for their insightful comments. Most of the research was carried out when E.V. was with Queen's University Belfast, UK.

Eleftherios Vlahakis is a Postdoctoral Researcher with the Division of Decision and Control Systems, KTH Royal Institute of Technology, Stockholm, Sweden. He was a Research Fellow (2020-2022) with the School of EEECS at Queen's University Belfast, UK. He obtained a Ph.D. (2020) in Control from the City, University of London, UK, an MSc degree (2014) in Energy Management and Production from the National Technical University of Athens, Greece, and an MEng degree (2007) in Electrical and Computer Engineering from Aristotle University of Thessaloniki, Greece. Dr. Vlahakis has worked (2009-2015) in various industrial roles in large-scale power networks and renewable energy systems. His research interests include distributed control, hybrid systems, and multi-agent control.

Raphaël Jungers is a Professor at UCLouvain, Belgium, currently on sabbatical leave at Oxford University. His main interests lie in the fields of Computer Science, Graph Theory, Optimisation and Control. He received a Ph.D. in Mathematical Engineering from UCLouvain (2008), and a M.Sc. in Applied Mathematics, both from the Ecole Centrale Paris, (2004), and from UCLouvain (2005).

He has held various invited positions, at the Université Libre de Bruxelles (2008-2009), at the Laboratory for Information and Decision Systems of the Massachusetts Institute of Technology (2009-2010), at the University of L´Aquila (2011, 2013, 2016), and at the University of California Los Angeles (2016-2017).

Prof. Jungers is a FNRS, BAEF, and Fulbright fellow. He has been an Editor at large for the IEEE CDC, Associate Editor for the IEEE CSS Conference Editorial Board, and the journals NAHS (2015-2016), Systems and Control Letters (2016-2017), IEEE Transactions on Automatic Control (2015-2020), Automatica (2020-). He was the recipient of the IBM Belgium 2009 award and a finalist of the ERCIM Cor Baayen award 2011. He was the co-recipient of the SICON best paper award 2013-2014, the HSCC2020 best paper award, and an ERC 2019 laureate.

Nikolaos Athanasopoulos received the Diploma and the Ph.D. degree in Electrical and Computer Engineering from the University of Patras, Patras, Greece, in 2004 and 2010, respectively. He is currently a Senior Lecturer with the School of Electronics, Electrical Engineering and Computer Science, Queen's University Belfast, U.K. His main interests lie in the control of cyber-physical systems, with a focus on hybrid systems, resource-aware control and set-based methods. Dr. Athanasopoulos is the recipient of an IKY and a Marie Curie fellowship and has held researcher positions in Eindhoven University of Technology, the Netherlands (2011-2014), and University of Louvain, Belgium (2015-2017).

Seán McLoone (S′94 – M′88 – SM′02) received an M.E. degree in Electrical and Electronic Engineering and a PhD in Control Engineering from Queen's University Belfast, Belfast, U.K. in 1992 and 1996, respectively.

He is currently a Professor and Director of the Energy Power and Intelligent Control Research Centre at Queen's University Belfast. His research interests are in Applied Computational Intelligence and Machine Learning with a particular focus on data based modelling and analysis of dynamical systems, with applications in advanced manufacturing informatics, energy and sustainability, connected health and assisted living technologies.

Prof. McLoone is a Chartered Engineer and Fellow of the Institution of Engineering and Technology. He is a Past Chairman of the UK and Republic of Ireland (UKRI) Section of the IEEE.