Optimal Resource Scheduling and Allocation 
under Allowable Over-Scheduling

Wei Ren, Eleftherios Vlahakis, Nikolaos Athanasopoulos and Raphaël M. Jungers

Abstract—This paper studies optimal scheduling and re-
source allocation under allowable over-scheduling. Formulating 
an optimisation problem where over-scheduling is embedded, 
we derive an optimal solution that can be implemented by 
means of a new additive increase multiplicative decrease 
(AIMD) algorithm. After describing the AIMD-like scheduling 
mechanism as a switching system, we show convergence of 
the scheme, based on the joint spectral radius of symmetric 
matrices, and propose two methods for fitting an optimal AIMD 
tuning to the optimal solution derived. Finally, we demonstrate 
the overall optimal design strategy via an illustrative example.

I. INTRODUCTION

With the advances in networking and hardware technology, 
distributed computing is increasingly adopted in many applica-
tions like multi-agent systems [1] and network systems [2]. 
Distributed computing systems have been proposed in differ-
ent fields [3]. In particular, scalability, reliability, information 
sharing/exchange from remote sources are main motivations 
for the users of distributed systems [4]. For distributed 
computing systems, many topics have been investigated [5]– 
[7], such as power management, request scheduling, resource 
allocation and system reliability, thereby resulting in different 
models, algorithms and software tools.

Among these extensively-studied topics, resource allo-
cation is one of most important problems due to its fundamental 
role in high performance of computing systems [3]. Resource 
allocation mechanisms include both scheduling of requests 
among different nodes and allocation of limited resources 
that are provisioned to requests entering computing systems. 
To solve resource allocation problems, many resource allo-
cation models and strategies have been proposed [3], such as 
proportional-share scheduling, market-based and auction-
based strategies. To avoid an excess capacity of resources, 
the over-scheduling phenomena are excluded in most existing 
strategies [6], [8], [9]. Over-scheduling phenomena may 
result in overuse of resources [10] and excess of predefined 
thresholds [6], [8], [9], leading to performance degradation, 
device damages, economic loss and even environmental 
threats. Nevertheless, in some scenarios, over-scheduling can 
be neither avoided in some scenarios nor is always harmful 
to computing systems [10]. For instance, over-scheduling is 
required in peak hours and may be caused by allocation 
strategies. Due to over-scheduling, the boundedness of the 
waiting time for each request is ensured and no additional 
techniques [8] are needed. To the best of our knowledge, 
there are few works on resource allocation under the over-
scheduling.

In this paper, motivated by the potential benefits of over-
scheduling, we deal with a resource allocation problem 
where a bounded over-scheduling is permitted. This effect-
ively allows us to implement an optimal scheduling strategy 
via a new variation of the additive increase multiplicative de-
crease (AIMD) algorithm which is known for its robustness 
properties and scalability. Here, we introduce an average-
based AIMD (A-AIMD) algorithm which tackles a schedul-
ing task from an average perspective and ensures bounded 
waiting time for all requests. Our proposed AIMD-based 
scheduling mechanism is motivated by recently introduced 
AIMD-like algorithms utilized for scheduling [8], [11]. First, 
the over-scheduling is embedded into an optimization prob-
lem, whose objective is to minimize the sum of the response 
time and cost of all computing nodes. The optimal strategy 
provides the bounds on request scheduling and resource al-
location without involving the A-AIMD mechanism. Hence, 
we highlight the inherent over-scheduling feature of the 
proposed A-AIMD algorithm that essentially guarantees that 
all requests are scheduled in finite time.

Although convergence and stability of typical AIMD 
schemes are well-studied in literature, see, e.g., [11], these 
properties need to be revisited here, primarily, due to a 
different triggering condition which is, to the best of our 
knowledge, novel in the context of event-triggered dis-
tributed systems. Due to its definition, the proposed A-
AIMD scheduling mechanism is intuitively transformed to a 
switching system the stability of which is shown by bound-
edness of the joint spectral radius of symmetric matrices 
[12]. Having established the convergence property of the 
scheduling scheme, we then highlight the relation between 
the magnitude of over-scheduling and the tuning parameters 
of the proposed A-AIMD setting. We finally propose two 
(complementary) methods for parameter design that can 
straightforwardly be used for optimal AIMD tuning.

The remainder of this paper is structured as follows. Section III formulates the considered problem. The optimal 
strategy is proposed in Section IV. The A-AIMD mechanism 
is addressed in Section V. A numerical example is given 
in Section VI. Conclusion and future work are presented in 

This work was supported by the H2020 ERC Consolidator Grant L2C 
(Grant 864017), the CHIST-ERA 2018 project DRUID-NET, the Walloon 
Region and the Innoviris Foundation.

W. Ren and R. M. Jungers are with ICTEAM institute, UCLouvain, 
1348 Louvain-la-Neuve, Belgium. E. Vlahakis and N. Athanasopoulos 
are with School of Electronics, Electrical Engineering and Computer 
Science, Queen’s University Belfast, Northern Ireland, UK. Email: 
wenren@uclouvain.be, e.vlahakis@qub.ac.uk, 
n.athanasopoulos@qub.ac.uk, 
raphael.jungers@uclouvain.be.

W. Ren and E. Vlahakis contributed equally.
Section VI

Notation. \( \mathbb{N} := \{0, 1, \ldots\}; \mathbb{N}^+ := \{1, 2, \ldots\}; \mathbb{R} := (-\infty, +\infty) \). \( \mathbb{R}^n \) is the \( n \)-dimensional Euclidean space. Given a function \( f : \mathbb{R}^n \rightarrow \mathbb{R} \) and \( x = (x_1, \ldots, x_n) \in \mathbb{R}^n \), we denote by \( \nabla f(x) \) the gradient of \( f \) with respect to \( x \).

Given a bounded set of matrices \( \Sigma = \{ A_1, \ldots, A_m \} \), the joint spectral radius of \( \Sigma \) is defined as \( \text{JSR}(\Sigma) = \lim_{k \rightarrow +\infty} \max_{\sigma \in \{1, \ldots, m\}^k} \| A_{\sigma_1} \cdots A_{\sigma_k} A_{\sigma_1} \|^{1/k} \).

II. Problem Formulation

We consider requests arriving at the dispatcher of a computing system illustrated in Fig. I. We model incoming requests as a continuous constant process with an arrival rate \( \lambda > 0 \); see also [11]. On their arrival, incoming requests are first queued in the dispatcher and then are distributed to \( n \) computing nodes. Each computing node scales its service capacity to minimize the processing time and cost.

To describe the number of all queued requests in the dispatcher at each time instant, the variable \( \delta(t) \geq 0 \) is introduced to denote the backlog of the dispatcher. For each computing node \( (i \in \mathcal{N} := \{1, \ldots, n\}) \), \( u_i(t) \geq 0 \) is the scheduling rate at which requests are distributed to the \( i \)-th computing node from the dispatcher at time \( t \); \( \gamma_i(t) > 0 \) is the service rate of the \( i \)-th computing node at time \( t > 0 \). If \( u_i(t) > \gamma_i(t) \), then the \( i \)-th computing node may store some requests before their execution. In this respect, similar to the dispatcher, the variable \( w_i(t) \geq 0 \) is introduced to denote the backlog of the \( i \)-th computing node. Since not all computing nodes need to be activated, \( u_i(t) \) and \( \gamma_i(t) \) can be zero. The next assumption is made.

Assumption I: For the considered computing system, the following holds.

1) For each \( i \in \mathcal{N} \), there exists \( \gamma_i^{\text{max}} > 0 \) such that \( u_i(t) < \gamma_i(t) \leq \gamma_i^{\text{max}} \) for all \( t > 0 \).

2) No constraints are imposed on the maximum backlog in the dispatcher.

In the first item of Assumption I, the first part of the inequality \( \gamma_i(t) \leq \gamma_i^{\text{max}} \) comes from the limited capacity of computing nodes, while \( u_i(t) < \gamma_i(t) \) can be imposed by the scheduling and allocation strategy to be designed. The second assumption implies that the dispatcher has sufficiently large capacity to store the incoming requests, which can be satisfied in practice; see, e.g., [13].

A. Average-based AIMD Mechanism

The classic AIMD mechanism [14, Ch. 14] consists of an additive increase (AI) phase and a multiplicative decrease (MD) phase. In the AI phase, the scheduling rates increase linearly with an additive rate \( \alpha_i > 0 \) until the capacity event \( \sum_{i=1}^n u_i = \lambda \) occurs. Immediately after the event, the MD phase is activated and the scheduling rate experiences an instantaneous decrease (i.e., a jump) with a multiplicative factor \( \beta_i \in (0, 1) \). In this way, the scheduling rates evolve as follows:

\[
 u_i(t) = \beta_i u_i(t_k) + \alpha_i (t - t_k), \quad \forall t \in (t_k, t_{k+1}], \tag{1}
\]

where \( t_k \) is the activation time of the MD phase and \( k \in \mathbb{N} \). That is, \( \lambda = \sum_{i=1}^n u_i(t_k) \) is the event-triggering condition for a typical AIMD scheme [14].

For the request scheduling and resource allocation as in Fig. I, the dynamics of \( \delta(t) \) and \( w_i(t) \) is derived below [15]:

\[
 \dot{\delta}(t) = \lambda - u_i(t), \quad \dot{w}_i(t) = u_i(t) - \gamma_i(t), \tag{2}
\]

where \( u_i(t) := \sum_{i=1}^n u_i(t) \). From (2) and Assumption I it holds that \( \dot{w}_i \leq 0 \), which implies the boundedness of the backlogs of all computing nodes. However, \( \sum_{i=1}^n u_i = \lambda \) cannot ensure the boundedness of \( \delta(t) \), since the over-scheduling is not allowed in the classic AIMD mechanism. To ensure that \( \delta(t) \) is finite and bounded for all \( t > 0 \), we propose an AIMD-like algorithm with a novel triggering condition. We call this average-based AIMD (A-AIMD).

Instead of \( \sum_{i=1}^n u_i = \lambda \), the jumps in the A-AIMD mechanism occur when \( \delta = 0 \), that is, no backlog in the dispatcher and thus no requests to be scheduled. Hence, the scheduling rates jump when \( \delta = 0 \), and the evolution of the scheduling rate is given in (1), where \( t_k \) is such that \( \delta(t_k) = 0 \). From (2), \( \delta(t_k) = 0 \) implies

\[
 \lambda T_k = \int_{t_k}^{t_k+1} \sum_{i=1}^n u_i(t) \, dt = \sum_{i=1}^n (\beta_i u_i(t_k) + 0.5 \alpha_i T_k) T_k, \tag{3}
\]

where \( T_k := t_{k+1} - t_k, \ k \in \mathbb{N} \). In this case, the event-triggered mechanism (ETM) for the MD phase is

\[
 t_{k+1} = \min \left\{ t > t_k : \lambda = \sum_{i=1}^n (\beta_i u_i(t_k) + 0.5 \alpha_i T_k) \right\}, \tag{4}
\]

which is the same as the one in [11]. We call it A-AIMD triggering condition as \( \beta_i u_i(t_k) + 0.5 \alpha_i T_k \) is the average share of each computing node between two successive jumps [14, Ch. 2].

Comparing with the classic AIMD mechanism, the A-AIMD mechanism allows for the over-scheduling. To be specific, if \( \lambda = \sum_{i=1}^n (\beta_i u_i(t_k) + 0.5 \alpha_i T_k) \) is satisfied such that the MD phase is activated, then \( \sum_{i=1}^n (\beta_i u_i(t_k) + \alpha_i T_k) > \lambda \), which implies that over-scheduling is inherent in the A-AIMD mechanism. However, over-scheduling may impose potential threats [6], and hence, its permissible magnitude should be subject to system constraints [10]. To avoid the unallowable over-scheduling, we assume

\[
 \sum_{i=1}^n u_i(t_k) = \sum_{i=1}^n (\beta_i u_i(t_k) + \alpha_i T_k) \leq \phi \lambda, \tag{5}
\]

where \( t_k \) is the activation time of the MD phase and \( k \in \mathbb{N} \). That is, \( \lambda = \sum_{i=1}^n u_i(t_k) \) is the event-triggering condition for a typical AIMD scheme [14].

For the request scheduling and resource allocation as in Fig. I, the dynamics of \( \delta(t) \) and \( w_i(t) \) is derived below [15]:

\[
 \dot{\delta}(t) = \lambda - u_i(t), \quad \dot{w}_i(t) = u_i(t) - \gamma_i(t), \tag{2}
\]
where $\phi > 1$ is given \textit{a priori}. By combining (4) and (5), we write the proposed A-AIMD system as follows:

$$u_i(t) = \beta_i u_i(t_k) + \alpha_i(t - t_k), \quad \forall t \in (t_k, t_{k+1}],$$  \hspace{1cm} (6a)

$$t_{k+1} = \min \left\{ t > t_k : \sum_{i=1}^{n} \beta_i u_i(t_k) + 0.5 \alpha_i t_k = \lambda \right\}.$$  \hspace{1cm} (6b)

In view of (6), the over-scheduling is clearly confined by the second item of the ETM which can be controlled by parameter $\phi$. One of the main contributions of this paper is to show that A-AIMD system (6a) is stable under triggering condition (6b). Hence, the problem to be studied in this paper is summarized below.

\textbf{Problem 1}: Consider the distributed computing system in Fig. II and the A-AIMD mechanism (6). Given a well-defined objective function, Assumption I and allowable over-scheduling, (1) determine the optimal strategy for both request scheduling and resource allocation; (2) establish the convergence property of the A-AIMD mechanism; (3) propose methods for AIMD tuning so that the derived optimal strategy is obtained by means of the A-AIMD mechanism.

III. OPTIMAL SCHEDULING AND ALLOCATION

To deal with Problem I we start with the first item by formulating and solving the optimization problem, the solution of which lays a foundation for the following two items in the next section.

A. Cost Function and Optimization Problem

To evaluate the scheduling and allocation, we first consider the QoS performance. Here, we propose the following average mean response time [16] as the QoS performance:

$$T := \sum_{i=1}^{n} \lambda_i^{-1} (\gamma_i - u_i)^{-1}. \hspace{1cm} (7)$$

Since $T$ needs to be non-negative and bounded, $\gamma_i > u_i$ is required, which also shows the reasonability of Assumption I. Under Assumption I, $T$ is convex with respect to $\gamma_i$ and $u_i$, respectively.

To evaluate cost of all computing nodes, we investigate the power consumption via the total service cost defined below:

$$C := \sum_{i=1}^{n} \lambda_i^{-1} u_i \varphi_i(\gamma_i), \hspace{1cm} (8)$$

where $\varphi_i(\gamma_i) : \mathbb{R}^+ \rightarrow \mathbb{R}^+$ is the service cost for each computing node. The service cost $\varphi_i(\gamma_i)$ includes the routing cost and the computing cost. The routing cost is assumed to be a constant, whereas the computing cost is related to and non-decreasing in the service rate $\gamma_i$. That is, the computing cost will increase if more computing resources are requested by each computing node. The computing cost includes the cost of the power used by each computing node and the memory costs required by each computing node. In this respect, for each computing node, its service cost is defined as follows:

$$\varphi_i(\gamma_i) := a_i \gamma_i^{b_i} + c_i \gamma_i + d_i, \quad \forall i \in \mathcal{N}, \hspace{1cm} (9)$$

where the first item is the power cost which is monomonal in the service rate; the second item is the cost of processor and storage memory; the last item is the routing cost; and $a_i, c_i, d_i > 0$ and $b_i > 1$. It is easy to check that $\varphi_i(\gamma_i)$ is convex, and its derivative with respect to $\gamma_i \geq 0$, i.e., $\varphi_i'(\gamma_i) := a_i b_i \gamma_i^{b_i-1} + c_i$, is positive and increasing with respect to $\gamma_i$. Since the service rate is upper bounded in practical systems, we can define $\varphi_i(\gamma_i) = \infty$ if $\gamma_i > \gamma_i^{\text{max}}$, see, e.g., [16].

Combining (7) and (8) yields the cost function $J := T + KC$, where $K > 0$ is a fixed weight. Hence, the following optimization problem can be formulated

$$\min \sum_{i=1}^{n} \lambda_i^{-1} u_i (\gamma_i - u_i)^{-1} + K \varphi_i(\gamma_i) \hspace{1cm} (10a)$$

s.t. $\sum_{i=1}^{n} u_i = \phi \lambda, \quad \phi > 1,$  \hspace{1cm} (10b)

$\gamma_i \leq \gamma_i^{\text{max}}, \quad 0 \leq u_i < \gamma_i, \quad \forall i \in \mathcal{N}, \hspace{1cm} (10c)$

where (10b) is from (5), and (10c) is from Assumption I.

B. Optimal Strategy under Over-Scheduling

The solution to (10) is derived in this subsection. Since not all computing nodes need to be activated, we first focus on how to choose the computing nodes to be activated. Based on the cost function (10a), we introduce the following variable for each computing node:

$$\theta_i := \phi \min_{0 < \gamma_i \leq \gamma_i^{\text{max}}} \{ \gamma_i^{-1} + K \varphi_i(\gamma_i) \}, \quad \forall i \in \mathcal{N}, \hspace{1cm} (11)$$

which can be treated as the price of each computing node [16]. To show this, if the dispatcher pays one euro per unit service time and $K$ euros per unit cost, then the price of each computing node is the expected total price that the dispatcher pays to this computing node. In this way, the dispatcher should choose the computing nodes with the lowest prices to ensure the minimization of the cost function. Hence, there exists a threshold for each computing node to decide whether this computing node is activated to schedule and process the requests. That is, each computing node is activated only when its price is below its threshold. For each $i \in \mathcal{N}$ and any $\theta \geq \phi K d_i$, we consider the equation with respect to $\gamma$:

$$K \phi \varphi_i(\gamma) + K \phi \varphi_i'(\gamma) = \theta. \hspace{1cm} (12)$$

In (12), $K \phi \varphi_i(\gamma) + K \phi \varphi_i'(\gamma)$ is lower bounded by $\phi K d_i$ and upper bounded due to $\gamma \leq \gamma_i^{\text{max}}$. Let $\gamma = g_i(\theta)$ be the solution to (12). Since $\varphi_i(\gamma)$ and $\varphi_i'(\gamma)$ are positive and increasing, $g_i(\cdot)$ is an increasing function. If $\theta$ exceeds certain bound $\theta_i^{\text{max}}$, which is related to $\gamma_i^{\text{max}}$, then no solution to (12) exists. In this case, let $g_i(\theta) = g_i(\theta_i^{\text{max}})$ for all $\theta \geq \theta_i^{\text{max}}$ and $i \in \mathcal{N}$.

\textbf{Theorem 1}: Consider the problem (10). Let Assumption I hold and $\min_{i \in \mathcal{N}} \{ \phi K d_i \} < \theta_1 \leq \ldots \leq \theta_n \leq \theta_{n+1} = \max_{i \in \mathcal{N}} \{ g_i^{-1}(\gamma_i^{\text{max}}) \}$ with the inverse function $g_i^{-1}$. The
the optimal solution satisfies the constraint qualification (GCQ) holds at

\[ \gamma_i - \gamma_i^{\max} = 0, \]

for \( n^* < i \leq n \),

\[ u_i^{\opt} = \begin{cases} \gamma_i - \sqrt{\frac{\phi \gamma_i^{\opt}}{\theta - K \gamma_i^{\opt}}}, & \text{for } 1 \leq i \leq n^*, \\ 0, & \text{for } n^* < i \leq n. \end{cases} \]  

(14)

where \( \theta \in (\theta_{n^*}, \theta_{n^*+1}] \) is such that \( \sum_{i=1}^{n} u_i^{\opt} = \phi \lambda \) and \( n^* := \arg\max_{\theta} \{ \theta : \sum_{i=1}^{n} (g_i(\theta) - 1/\sqrt{K} \gamma_i^{\opt}(g_i(\theta))) \leq \phi \lambda \}. \)

Proof: The Lagrangian for (10) is defined as

\[ \mathcal{L} := \sum_{i=1}^{n} \lambda_{i}^{-1} u_i ((\gamma_i - u_i)^{-1} + K \gamma_i^{\opt}(\gamma_i)) - \theta (\sum_{i=1}^{n} \phi \lambda \gamma_i^{\max} - 1) - \sum_{i=1}^{n} b_i u_i - \sum_{i=1}^{n} h_i (\gamma_i - u_i) + \sum_{i=1}^{n} k_i (\gamma_i - \gamma_i^{\max}), \]

where \( \lambda, b_i, h_i, k_i \in \mathbb{R}^+ \) are Lagrange multipliers. Since the cost function is not convex with respect to \((\gamma_i, u_i)\), the problem (10) is not convex. To derive the optimal solution, we show the satisfaction of the KKT condition for the optimal solution first, and then the uniqueness of the optimal solution.

Since not all computing nodes need to be activated, the set \( \mathcal{N} \) is partitioned into the inactivated part \( \mathcal{I} \subseteq \mathcal{N} \) and the activated part \( \mathcal{A} \subseteq \mathcal{N} \). Thus, \( \mathcal{I} \cap \mathcal{A} = \emptyset, \mathcal{I} \cup \mathcal{A} = \mathcal{N} \), and \( u_i = \gamma_i = 0 \) for all \( i \in \mathcal{I} \). In this respect, the optimization problem (10) is rewritten as

\[ \min_{u_i} \sum_{i=1}^{n} \lambda_{i}^{-1} u_i ((\gamma_i - u_i)^{-1} + K \gamma_i^{\opt}(\gamma_i)) \]

\[ \text{s.t.} \quad G_i(u), \gamma_i := \begin{cases} \gamma_i - \gamma_i^{\max} \leq 0, & \forall i \in \mathcal{A}, \\ G_i(u), \gamma_i := u_i - \gamma_i \leq 0, & \forall i \in \mathcal{A}, \end{cases} \]

(15a-e)

where \( u := (u_1, \ldots, u_n) \in \mathbb{R}^n \) and \( \gamma := (\gamma_1, \ldots, \gamma_n) \in \mathbb{R}^n \). From (15), \( \nabla G_i(u, \gamma) \) is only related to \( u \) and linearly independent, whereas \( \nabla G_2(u, \gamma) := (\nabla G_{21}(u, \gamma), \ldots, \nabla G_{2n}(u, \gamma)) \) is only related to \( \gamma \) and linearly independent. Hence, we can easily check that \( \nabla G_1(u, \gamma) \) and \( \nabla G_2(u, \gamma) \) with \( i \in \mathcal{A} \) are linearly independent. From [17, Def. 4.1] and [17, Thm. 4.3], the Guignard constraint qualification (GCQ) holds at \((u, \gamma)\), and further from [18, Thm. 6.1.4], the optimal solution satisfies the following KKT conditions:

\[ \frac{\partial \mathcal{L}}{\partial u_i} = \frac{\gamma_i}{(\gamma_i - u_i)^2} + \frac{\phi \lambda \gamma_i^{\opt}}{\phi \lambda} - b_i + h_i = 0, \]

(16a)

\[ \frac{\partial \mathcal{L}}{\partial \gamma_i} = -\frac{u_i}{(\gamma_i - u_i)^2} + \frac{K u_i \gamma_i^{\opt}}{\lambda} - h_i + k_i = 0, \]

(16b)

\[ \sum_{i=1}^{n} \phi \lambda_{i}^{-1} u_i = 1, \quad \sum_{i=1}^{n} b_i u_i = 0, \]

(16c)

\[ \sum_{i=1}^{n} h_i (\gamma_i - u_i) = 0, \quad \sum_{i=1}^{n} k_i (\gamma_i - \gamma_i^{\max}) = 0. \]

(16d)

From (16c-e) and \( u_i < \gamma_i, h_i \equiv 0 \). Then, from (16a),

\[ u_i^{\opt} = \max \left\{ 0, \gamma_i^{\opt} - \sqrt{\frac{\phi \gamma_i^{\opt}}{\theta - K \phi \gamma_i^{\opt}}} \right\}, \]

(17)

where \( \gamma_i^{\opt} \) is the optimal service rate. If \( \gamma_i < \gamma_i^{\max} \), then \( k_i = 0 \) from (16d), and further from (16a),

\[ u_i^{\opt} ((\gamma_i^{\opt} - u_i^{\opt}) - 2 - K \phi \gamma_i^{\opt}) = 0. \]

(18)

If \( u_i^{\opt} > 0 \), then \( b_i = 0 \) from (16c), and from (17)-(18),

\[ \gamma_i^{\opt} (\gamma_i^{\opt} - u_i^{\opt} - 2) = \phi - \theta - K \phi \gamma_i^{\opt}, \]

(19)

which shows \( \gamma_i^{\opt} = g_i(\theta) \) from (13). If \( u_i^{\opt} = 0 \) for some \( i \in \mathcal{N} \), then the optimal value of \( \gamma_i^{\opt} \) has no effects on the cost function in (10a), and can be chosen arbitrarily from \([0, \gamma_i^{\max}] \). In this case, we can still choose \( \gamma_i^{\opt} = g_i(\theta) \), whereas \( b_i \) needs to be chosen appropriately to ensure the KKT conditions (16a-16d). If \( \gamma_i = \gamma_i^{\max} \), then \( k_i \geq 0 \) can be chosen arbitrarily. In this case, we can follow the above argument to derive the same optimal values. On the other hand, if \( u_i^{\opt} > 0 \), then \( b_i = 0 \) and from (17), \( \theta > \phi \gamma_i^{\opt} + K \phi \gamma_i^{\opt} \geq \theta_i \), where \( \theta_i \) is defined in (11). If \( u_i^{\opt} = 0 \), then from (17) and (12),

\[ \theta + \phi \lambda b_i \leq \phi g_i(\theta) (1 - K \phi g_i(\theta)) - g_i^{-2}(\theta). \]

(20)

From (16b), (20) holds if either \( g_i(\theta) = 0 \) or \( K \phi g_i(\theta) - g_i^{-2}(\theta) \leq 0 \). If \( K \phi g_i(\theta) - g_i^{-2}(\theta) \leq 0 \), then \( g_i(\theta) \leq \bar{\gamma}_i \), where \( \bar{\gamma}_i = g_i(\theta) \) is from [8, Lem. 1]. Since \( g_i(\cdot) \) is increasing, \( g_i(\theta) \leq g_i(\theta_i) \) implies \( \theta \leq \theta_i \). Hence, in the derived optimal strategy, the computing nodes are chosen via the increasing order of \( \theta_i \). In addition, \( \sum_{i=1}^{n} u_i^{\opt} = \phi \lambda \) needs to be satisfied, thereby resulting in the constraint on \( n^* \).

Next, we show the uniqueness of the derived solution via the contradiction. Let \((\hat{\gamma}_i, \hat{u}_i)\) and \((\bar{\gamma}_i, \bar{u}_i)\) be two different solutions to (10). Since \( \sum_{i=1}^{n} \hat{u}_i = \sum_{i=1}^{n} \bar{u}_i = \phi \lambda \), we have \( \hat{u}_i \neq \bar{u}_i \) for all \( i \in \{1, \ldots, \max\{\hat{n}, \bar{n}\}\} \), which further implies from (14) that \( \hat{\gamma}_i \neq \bar{\gamma}_i \) for all \( i \in \{1, \ldots, \max\{\hat{n}, \bar{n}\}\} \). However, for the equation \( \sum_{i=1}^{n} (x_i - \sqrt{\frac{\phi \gamma_i}{\theta - K \phi \gamma_i}}) = \phi \lambda \) with \( \phi > 1 \) and \( \lambda > 0 \), its solution is unique from [19], and thus \( \hat{\gamma}_i = \bar{\gamma}_i \) for all \( i \in \{1, \ldots, n\} \), which results in a contradiction. Therefore, the solution to the problem (10) is unique, which completes the proof.

Theorem III shows how to determine scheduling and service rates in the over-scheduling case, which is different from our previous work [8] where over-scheduling is not considered. In addition, Theorem III provides bounds on scheduling and service rates related to over-scheduling the allowable magnitude of which can be controlled by the tuning parameter \( \phi > 1 \).

IV. A-AIMD MECHANISM

In the derived optimal strategy, the evolution of the scheduling rates is not involved, and the effect of \( \phi > 1 \) on
the A-AIMD mechanism deserves further study. Motivated
by these observations, we address the A-AIMD mechanism
in this section. In particular, the convergence property of
the A-AIMD mechanism is derived, which is further combined
with the optimal strategy to design the AIMD parameters.

A. Convergence under Over-Scheduling

To show convergence, we first transform the A-AIMD
mechanism \( \mathbf{G} \) into a switched system whose stability guar-
antees that scheduling rates converge to certain fixed points.

We define the following functions:

\[
G(u, v) = \max\{G_1(u, v), G_2(v)\},
\]

\[
G_1(u, v) = 0.5 \sum_{i=1}^{\Sigma} (\beta_i u_i + v_i), \quad G_2(v) = \phi^{-1} \sum_{i=1}^{\Sigma} v_i,
\]

where \( u = (u_1, \ldots, u_n) \), \( v = (v_1, \ldots, v_n) \). Therefore, the ETM (25) is rewritten as

\[
t_{k+1} = \min\{t > t_k : G(u(t_k), u(t)) = \lambda\},
\]

Let \( T_1(t_k) \geq 0 \) (\( T_2(t_k) \geq 0 \)) be the inter-event period
when \( G_1 = \lambda \) (\( G_2 = \lambda \)). If \( G(u(t_k), u(t_{k+1})) = \)
\( G_1(u(t_k), u(t_{k+1})) \), then \( T_1(t_k) \leq T_2(t_k) \) and \( T_k = T_1(t_k) \), where \( T_k \) is defined in (3). In this case, combining
\( G_1(u(t_k), u(t_{k+1})) = \phi \) and (4) yields

\[
T_1(t_k) = \frac{2\lambda - \sum_{i=1}^{n} \beta_i u_i(t_k)}{\sum_{i=1}^{n} \alpha_i}, \quad u_i(t_{k+1}) = \beta_i u_i(t_k) + \alpha_i T_1(t_k).
\]

If \( G(u(t_k), u(t_{k+1})) = G_2(u(t_{k+1})) \), then \( T_k = T_2(t_k) < T_1(t_k) \), and \( T_2(t_k) \) is defined via \( G_2(u(t_{k+1})) = \lambda \). That is,

\[
T_2(t_k) = \frac{\phi \lambda - \sum_{i=1}^{n} \beta_i u_i(t_k)}{\sum_{i=1}^{n} \alpha_i}, \quad u_i(t_{k+1}) = \beta_i u_i(t_k) + \alpha_i T_2(t_k).
\]

In view of (23), the aggregate dynamics of the A-
AIMD mechanism (6) can be expressed by the following
state-dependent switched system.

\[
u(t_{k+1}) = \begin{cases} A_1 u(t_k) + B_1 \quad \text{if } \beta^T u(t_k) \geq (2 - \phi)\lambda \\ A_2 u(t_k) + B_2 \quad \text{if } \beta^T u(t_k) < (2 - \phi)\lambda \end{cases}
\]

with \( u(t_k) = (u_1(t_k), \ldots, u_n(t_k)) \in \mathbb{R}^n \), \( A_1 = \text{diag} \{\beta\} - 2\alpha^T \beta^T \), \( A_2 = \text{diag} \{\beta\} - \frac{\alpha}{\sum_{i=1}^{n} \alpha_i} \alpha^T \), \( B_1 = \frac{2\lambda}{\sum_{i=1}^{n} \alpha_i} \), and \( B_2 = \frac{2\lambda}{\sum_{i=1}^{n} \alpha_i} \), where \( \beta = (\beta_1, \ldots, \beta_n) \in \mathbb{R}^n \), \( \alpha = (\alpha_1, \ldots, \alpha_n) \in \mathbb{R}^n \) and diag(\( \cdot \)) is the diagonal operator.

Clearly, (25) is a discrete-time state-dependent switched system with two modes where the ETM (22) is embedded into the switching conditions. Since \( \beta^T u(t_k) \geq 0 \), then \( \phi < 2 \) is needed to ensure that the condition \( \beta^T u(t_k) < (2 - \phi)\lambda \) in (25) is reasonable. Therefore, we have \( \phi \in (1, 2) \).

Before showing the convergence of the scheduling rates via (25), we first present a proposition to show the stability properties of matrices \( A_1 \) and \( A_2 \) as defined in (25).

**Proposition 1:** Let matrices \( A_1, A_2 \) be defined as in (25) and the matrix set be \( \Sigma = \{A_1, A_2\} \). The joint spectral radius of \( \Sigma \) is defined in (3). In this case, combining (23) and (25) yields

\[
\sum_{i=1}^{n} \frac{\alpha_i}{1 - \beta_i \alpha_i} = 1 + \sum_{i=1}^{n} \frac{\alpha_i}{1 - \beta_i \alpha_i} = \phi.
\]
Combining (29) with (28) yields \( T_1^* = T_2^* = T^* \) with \( T^* \) defined in (27). Hence, \( \bar{u}_{\text{con}} = \bar{u}_{\text{con}} = u_{\text{con}} \) with \( u_{\text{con}} := (u_{\text{con}}^1, \ldots, u_{\text{con}}^n) \). Next, we show that the system (25) converges to the fixed point as in (27). We define the error \( e(t_k) := u(t_k) - u_{\text{con}} \), whose dynamics is given by

\[
e(t_{k+1}) = \begin{cases} A_1 e(t_k) & \text{if } \beta^T e(t_k) \geq 0 \\ A_2 e(t_k) & \text{if } \beta^T e(t_k) < 0. \end{cases}
\]

(30)

From Proposition \([11, \text{Cor. 1.1.}]\), the system (30) is asymptotically stable, which means that the origin is a fixed point for (30), i.e., \( \lim_{k \to \infty} e(t_k) = 0 \), or \( \lim_{k \to \infty} u(t_k) = u_{\text{con}} \) with \( u_{\text{con}} \) given in (27), which completes the proof of the first item.

Next, we show the second item. From (2), the backlog \( \delta(t_{k+1}) \) is given below:

\[
\delta(t_{k+1}) = \delta(t_k) + \int_{t_k}^{t_{k+1}} (\lambda - u(t)) dt
\]

\[
= \delta(t_k) + \lambda T_k - 0.5(\beta^T u(t_k) + 1^T u(t_{k+1})) T_k.
\]

In Mode 1, we have from the event-triggering condition that \( 0.5(\beta^T u(t_k) + 1^T u(t_{k+1})) = \lambda \), which implies that \( \delta(t_{k+1}) = \delta(t_k) \). In Mode 2, the event-triggering condition results in \( 1^T u(t_{k+1}) = \phi \lambda \), and thus

\[
\delta(t_{k+1}) = \delta(t_k) + \frac{(2 - \phi)\lambda - \beta^T u(t_k)}{2} T_k,
\]

which, from (29), becomes

\[
\delta(t_{k+1}) = \delta(t_k) + \frac{\beta^T u_{\text{con}} - \beta^T u(t_k)}{2} T_k
\]

\[
= \delta(t_k) - 0.5\beta^T e(t_k) T_k.
\]

Therefore, the dynamics of \( \delta(t_{k+1}) \) is

\[
\delta(t_{k+1}) = \begin{cases} \delta(t_k) & \text{if } \beta^T e(t_k) \geq 0 \\ \delta(t_k) - 0.5\beta^T e(t_k) T_k & \text{if } \beta^T e(t_k) < 0. \end{cases}
\]

(31)

In addition, the asymptotic stability of the system (25) implies that there exists \( L \geq 1 \) such that

\[
\|e(t_k)\| \leq L(\|S\|k\|e(t_0)\|) \leq L\beta^T e(t_k),
\]

(34)

where the second inequality holds from Proposition \([11, \text{Cor. 1.1.}]\). Combining (33) and (34) yields

\[
\delta(t_{k+1}) \leq \delta(t_0) + 0.5\beta_{\text{max}} T_{\text{max}} \sum_{l=0}^{k} L\beta^T e(t_l)
\]

\[
\leq \delta(t_0) + 0.5\|e(t_0)\| T_{\text{max}} L \sum_{l=0}^{k} \beta_{\text{max}} T_{\text{max}}
\]

\[
\leq \delta(t_0) + \frac{L\beta_{\text{max}} T_{\text{max}}\|e(t_0)\|}{2(1 - \beta_{\text{max}})}.
\]

Since both \( \delta(t_0) \) and \( \|e(t_0)\| \) are bounded at the initial time, we have the boundedness of \( \delta \), which hence completes the proof of the second item.

In Theorem 2, the convergence property is preserved for the A-AIMD mechanism under (26), which can be treated as the constraints on the AIMD parameters \( \alpha_i, \beta_i \) and will be discussed in the next subsection.

**B. Tuning of AIMD Parameters**

From Theorem 2 we have that if the AIMD parameters take values in the set induced by (26) the resulting system (25) is stable with a unique fixed point. In this section, we combine the solutions in Theorems 1 and 2 to ensure the convergence of all scheduling rates to the optimal scheduling rates, thereby combining the optimal strategy with the A-AIMD mechanism.

From Theorem 1 \( u_i^{\text{opt}} = \gamma_i^{\text{opt}} = 0 \) for all \( i \in \mathcal{A} \), and thus we set \( \alpha_i = \beta_i = 0 \) directly for all \( i \in \mathcal{A} \). Next, we investigate the case of \( i \in \mathcal{A} \). For all \( i \in \mathcal{A} \), there are two constraints on the scheduling rates: \( u_{\text{con}} = u_i^{\text{opt}} \) from our goal and \( u_i^{\text{opt}} < \gamma_i^{\text{opt}} \) from Assumption 1. Hence, there exist two cases. The first case is that \( (1 - \beta_i)^{-1} \alpha_i T^* \geq \gamma_i^{\text{opt}} \). For this case, from Assumption 1 and \( u_i^{\text{opt}} = u_i^{\text{con}} \), we have \( u_i^{\text{opt}} = \gamma_i^{\text{opt}} - \epsilon \), where \( \epsilon > 0 \) is sufficiently small to avoid \( u_i^{\text{opt}} = \gamma_i^{\text{opt}} \). From (14), \( K\varphi_i^{(\gamma_i^{\text{opt}})} = \epsilon^2 \). Since \( \epsilon > 0 \) is arbitrarily small, we have from (9) and the derivative of \( \varphi_i \), that \( \gamma_i^{\text{opt}} \) needs to be sufficiently large, which contradicts with the boundedness of \( \gamma_i \). Therefore, the first case is unreasonable, and we only need to consider the second case: \( (1 - \beta_i)^{-1} \alpha_i T^* < \gamma_i^{\text{opt}} \). In the second case, from (28), \( u_i^{\text{opt}} = (1 - \beta_i)^{-1} \alpha_i T^* \), which is rewritten as

\[
(1 - \beta_i) u_i^{\text{opt}} = \alpha_i T^*, \quad \forall i \in \mathcal{A}.
\]

(35)

Due to (26) and \( T^* \) in (35), the parameters \( \alpha_i \) and \( \beta_i \) affect each other. We next propose two methods to design the AIMD parameters.

1) Linear polynomial based design: Let \( x_i := \alpha_i T^*, y_i := \beta_i \). Hence, (35) equals to \( x_i + u_{\text{con}}^{\text{opt}} y_i = u_i^{\text{opt}} \).

Since \( \phi > 1 \) is known a priori, we have from (31) and (35),

\[
\sum_{i=1}^{n} x_i = 2(\phi - 1)\lambda, \quad \sum_{i=1}^{n} (0.5x_i + u_{\text{con}}^{\text{opt}} y_i) = \lambda.
\]

(36)
where, the former one is equivalent to the constraint in (26)-(27) on $x_i$. All these constraints can be written into a unified equation form $Az = B$ with $z := (x, y, \phi)$, $x := (x_1, \ldots, x_n)$, $y := (y_1, \ldots, y_n)$ and

$$A := \begin{bmatrix} I_{n \times n} & \text{diag}(u^\text{opt}) & 0 \\ 1^T & 0^T & -2\lambda \\ 0.51^T & u^\text{opt} & 0 \end{bmatrix}, \quad B := \begin{bmatrix} u^\text{opt} \\ -2\lambda \\ \lambda \end{bmatrix}, \quad (37)$$

where $u^\text{opt} := (u_1^\text{opt}, \ldots, u_n^\text{opt}) \in \mathbb{R}^n$ and $0 \in \mathbb{R}^n$ is the vector whose components are 0. Since $\phi > 0$ is known a priori, $Az = B$ is an equation with a known variable. From [20], we can check that $A \in \mathbb{R}^{(n+2) \times (2n+1)}$ is full-rank and its null space is non-trivial. Hence, the solution to $Az = B$ is existent and non-unique. Different approaches, e.g., the method of least squares, can be applied to find feasible solutions.

2) Optimization-based design: Since the above method cannot show the uniqueness of the solution, another design method is based on the optimization theory. Here the goal is to minimize the Euclidean norm of the AIMD parameters under (35)-(36). The optimization problem is formulated below:

$$\min_{x, y, \phi} \sum_{i=1}^{n} (x_i^2 + y_i^2) + \phi^2 \quad \text{s.t. (35)-(36), } \quad u^\text{opt} < \lambda, \quad x_i > 0, \quad y_i \in (0, 1).$$

From (35), the cost function in (38) can be rewritten as a function of $y$ and $\phi$. Similar to the first method, $\phi$ can be involved in the cost function such that we have the co-design of the optimal over-scheduling and AIMD parameters. If $\phi$ is fixed, then the cost function is only related to $y$. Since the problem (38) is convex, the solution is unique, which hence shows the advantages of the optimization-based method.

V. NUMERICAL RESULTS

Consider a computing system with 3 heterogeneous computing nodes with different service capacities and power consumptions. To be specific, let the coefficients in (10) be $a_1 = 0.1, a_2 = 0.2, a_3 = 0.4, b_1 = b_2 = b_3 = 2, c_1 = 0.3, c_2 = 0.6, c_3 = 0.9, d_1 = 1, d_2 = 2, d_3 = 3$. In addition, $\gamma_1^\text{max} = 6, \gamma_2^\text{max} = 9, \gamma_3^\text{max} = 11$ and the weight $K = 1.2$. Therefore, from (11), $\theta_1 = 2.6373\phi$, $\theta_2 = 4.3524\phi$, $\theta_3 = 6.0133\phi$. Hence, the lightweight computing node (i.e., Node 1) is always activated. Moreover, the upper bound of $\theta_i$ is $\theta_4 = 201.6\phi$. With different $\phi \in [1, 2]$, the number of the activated computing nodes is presented in Fig. 3. Let $\theta = 13.648$ in (14)-(15), $\phi = 1.4$ and $\lambda = 5.5$, and from Theorem 1 the optimal rates are $u_1^\text{opt} = 4.1619, u_2^\text{opt} = 2.4024, u_3^\text{opt} = 1.4182, \gamma_1^\text{opt} = 4.9647, \gamma_2^\text{opt} = 3.0770, \gamma_3^\text{opt} = 1.9660$. The A-AIMD mechanism and over-scheduling are studied below.

First, we assume that the AIMD parameters are known. Let $\alpha_1 = 0.3, \alpha_2 = 0.5, \alpha_3 = 0.8, \beta_1 = 0.5, \beta_2 = 0.35, \beta_3 = 0.15$. From Theorem 2 $T^* = 2.75, u_1^\text{con} = 1.65, u_2^\text{con} = 2.1154, u_3^\text{con} = 2.5882$; see Fig. 2. It is easy to check that $\lambda < u_1^\text{con} + u_2^\text{con} + u_3^\text{con} \leq \phi$. Next, we consider the known over-scheduling case. From Section V-B, the design of the AIMD parameters is formulated into an optimization problem (35), and thus we have $\alpha_1^* T^* = 1.7662, \alpha_2^* T^* = 1.5559, \alpha_3^* T^* = 1.0780, \beta_1 = 0.5584, \beta_2 = 0.3197, \beta_3 = 0.1867$. That is, for each $i \in \{1, 2, 3\}$, $\beta_i$ is fixed, whereas $\alpha_i^*$ and $T^*$ can be determined via the user requirement. The feasible choice is presented in Fig. 4. For instance, a feasible choice is $\alpha_1^* = 0.4750, \alpha_2^* = 0.4185, \alpha_3^* = 0.2899$ and $T^* = 3.7180$. Finally, if $\phi$ is embedded into (38), then we derive the optimal coefficient $\phi^* = 1.4514$ and the optimal AIMD parameters $\alpha_1^* T^* = 2.0799, \alpha_2^* T^* = 1.7086, \alpha_3^* T^* = 1.1765, \beta_1^* = 0.5003, \beta_2^* = 0.2888, \beta_3^* = 0.1705$, which thus guarantee the optimal strategy in Theorem 1. In particular, if $T^* = 1.5$, then $\alpha_1^* = 1.3866, \alpha_2^* = 1.1391, \alpha_3^* = 0.7843$; see the bold points in Fig. 4.

VI. CONCLUSION

In this paper, we studied the resource allocation problem of computing systems under over-scheduling phenomena. We proposed an optimisation problem and derived an optimal scheduling and allocation strategy. To realize the optimal scheduling solution under over-scheduling, we proposed a feedback mechanism based on a new AIMD-like algorithm whose stability was proved by means of the boundedness of the joint spectral radius associated with an equivalent switched system. Finally, two methods for tuning AIMD parameters were proposed to fit an optimal AIMD-based mechanism to the optimal solution.

REFERENCES

[1] M. Woolridge and M. J. Wooldridge, Introduction to Multiagent Systems. John Wiley & Sons, Inc., 2001.
[2] A. D. Kshemkalyani and M. Singhal, Distributed Computing: Principles, Algorithms, and Systems. Cambridge University Press, 2011.
[3] H. Hassan, S. U. R. Malik, A. Hameed, S. U. Khan, G. Bickler, N. Min-Allah, M. B. Qureshi, L. Zhang, W. Yongji, N. Ghani et al., "A survey on resource allocation in high performance distributed computing systems," Parallel Computing, vol. 39, no. 11, pp. 709–736, 2013.
Fig. 4. Illustration of the relation among $\alpha_i$, $T^\ast$ and $\beta_i$. The curves with light colors are from (26). The curves with blue, red and green colors are a feasible solution for the case where $\phi = 1.4$. The curves with dark colors are for the optimal case $\phi^* = 1.4514$.

[4] Y. Amir et al., “An opportunity cost approach for job assignment in a scalable computing cluster,” IEEE Trans. Parallel. Distrib. Syst., vol. 11, no. 7, pp. 760–768, 2000.

[5] B. Y. Kimura, D. C. Lima, and A. A. Loureiro, “Packet scheduling in multipath TCP: Fundamentals, lessons, and opportunities,” IEEE Systems Journal, vol. 15, no. 1, pp. 1445–1457, 2020.

[6] A. Pentelas, G. Papathanail, I. Fotoglou, and P. Papadimitriou, “Network service embedding across multiple resource dimensions,” IEEE Netw. Serv. Manag., vol. 18, no. 1, pp. 209–223, 2020.

[7] J. G. Herrera and J. F. Botero, “Resource allocation in NFV: A comprehensive survey,” IEEE Trans. Netw. Serv. Manag., vol. 13, no. 3, pp. 518–532, 2016.

[8] W. Ren, E. Vlahakis, N. Athanasopoulos, and R. Jungers, “Optimal resource scheduling and allocation in distributed computing systems (accepted),” American Control Conference, 2022.

[9] A. C. Luna, N. L. Diaz, M. Graells, J. C. Vasquez, and J. M. Guerrero, “Mixed-integer-linear-programming-based energy management system for hybrid PV-wind-battery microgrids: Modeling, design, and experimental verification,” IEEE Trans. Power Electron., vol. 32, no. 4, pp. 2769–2783, 2017.

[10] M. Pantazoglou, G. Taortzakis, and A. Delis, “Decentralized and energy-efficient workload management in enterprise clouds,” IEEE Trans. Cloud Comput., vol. 4, no. 2, pp. 196–209, 2015.

[11] E. Vlahakis, N. Athanasopoulos, and S. McLoone, “AIMD scheduling and resource allocation in distributed computing systems,” in Proc. IEEE Conf. Decis. Control. IEEE, 2021.

[12] R. Jungers, The Joint Spectral Radius: Theory and Applications. Springer Science & Business Media, 2009, vol. 385.

[13] Amazon, “Unlimited cloud storage from amazon,” Available at: https://www.amazon.com/clouddrive/home, 2015.

[14] M. Corless, C. King, R. Shorten, and F. Wirth, AIMD Dynamics and Distributed Resource Allocation. SIAM, 2016.

[15] F. L. Presti, Z.-L. Zhang, and D. Towsley, “Bounds, approximations and applications for a two-queue GPS system,” in IEEE Conference on Computer Communications, vol. 3. IEEE, 1996, pp. 1310–1317.

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

[17] T. Hoheisel and C. Kanzow, “On the Abadie and Guignard constraint qualifications for mathematical programmes with vanishing constraints,” Optimization, vol. 58, no. 4, pp. 431–448, 2009.

[18] M. S. Bazaraa and C. M. Shetty, Foundations of Optimization. Springer Science & Business Media, 1976.

[19] T. Fujisawa and E. Kuh, “Some results on existence and uniqueness of solutions of nonlinear networks,” IEEE Trans. Circuit Theory, vol. 18, no. 5, pp. 501–506, 1971.

[20] G. Strang, Linear Algebra and Its Applications, 4th Edition. Wellesley-Cambridge Press, 2005.