

Article

A Novel Model for Online Scheduling of Approximation Jobs [†]

Qi Li ^{1,*}, Xiaolei Wang ¹, Shuo Wen ¹, Wei Du ¹, Li Mao ¹ and Lijun Cai ²

¹ Department of Network Information Security, Guangdong Police College, Guangzhou 510440, China; 19950355@gdppla.edu.cn (X.W.); wen.shuo@foxmail.com (S.W.); weiduzxx@163.com (W.D.); 20040361@gdppla.edu.cn (L.M.)

² Computer Science and Electronic Engineering, Hunan University, Changsha 410012, China; ljcai@hnu.edu.cn

* Correspondence: qili@hnu.edu.cn

[†] This paper is an extended version of our paper published in ISPA 2025 (Li, Q.; Wen, S.; Wang, X.; Du, W.; Cai, L.; Xu, H. Online Job Scheduling for Profit Maximization in a Heterogeneous Cluster. In *23rd IEEE International Symposium on Parallel and Distributed Processing with Applications*; IEEE: New York, NY, USA, 2025) which is held in Shenyang of China.

Abstract

Approximation jobs are widely deployed on Amazon EC2, which compute partial task segments to obtain useful results. For such jobs, maximizing total profit is the primary goal, where profit equals the sum of job utilities minus the total machine costs. Unfortunately, maximizing the total profit of approximation jobs is an NP-hard problem. This problem is further complicated by online job arrivals and heterogeneous resource demands across different tasks. This work builds an optimization framework that clearly characterizes job utility and machine costs to resolve this problem. Within this framework, we propose an efficient dual algorithm for job scheduling. The proposed method leverages the dual-fitting approach to measure algorithm performance by analyzing the primal and dual objective growth at each step. This work proves that our algorithm achieves a constant competitive ratio. The results from the trace-driven simulations demonstrate that our algorithms consistently outperform these baselines across various metrics.

Keywords: online job assignments; maximizing profit; heterogeneous resources; dual-fitting; competitive performance

1. Introduction

This paper is an extended version of our paper published in ISPA 2025 [1] which is held in Shenyang of China.

By the increasing volume and complexity of data in today's world, parallel and distributed computing has become crucial for supporting large-scale data analytics. Amazon EC2 offers native frameworks that allow users to deploy and distribute approximation jobs over thousands of virtual machines [2]. Such jobs are composed of abundant tiny subtasks and only demand partial execution to yield usable results [3]. High job utility is an essential metric for data analytics running on computing clusters. Cluster schedulers typically allocate shared resources to multiple jobs at the same time to maximize summed utility. Job utility is formalized as a concave function dependent on the volume of completed subtasks [4]. Users must bear rental costs for the EC2 instances they occupy within the cluster to support analytic computation [5]. Reducing total expenditure is also vital to elevate resource efficiency in cluster environments. Per-machine cost is expressed as a convex function of the tasks finished on each instance. We define total profit as the aggregate utility of all running jobs minus the overall cost brought by leased machines. Ref. [6]



Academic Editor: Massimiliano Caramia

Received: 20 May 2026

Revised: 22 June 2026

Accepted: 23 June 2026

Published: 2 July 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

targets heterogeneous accelerator scheduling for industrial cloud platforms built on the AWS/EC2 ecosystem and standard Kubernetes orchestration. Ref. [7] studies two critical layers of heterogeneity in end-edge-cloud collaborative clusters: hierarchical infrastructure heterogeneity across three tiers and multi-dimensional heterogeneous computing/network resources within each tier. Ref. [8] adopts competitive analysis as the primary theoretical tool to evaluate online auction performance against the offline optimal benchmark.

Driven by the growing volume of cloud applications, modern large-scale clusters process vast streams of online jobs across tens of thousands of deployed instances. Three key challenges render this scheduling problem intractable. For one thing, the system suffers from dual heterogeneity across hardware and workloads: jobs vary in execution length and carry heterogeneous utility formulations [9,10]. For another, online job assignment is inherently an integer programming problem [11]. The profit-maximizing online scheduling problem (future job arrivals remain unknown in advance) has been proven NP-hard. Last, approximation jobs only demand partial task execution to deliver acceptable outputs [12]. Periodically assessing profit increments per time slot is essential for performance optimization. This periodic assessment enables the scheduler to strategically select tasks for execution so as to maximize total system profit.

In online scheduling, the traditional model does not characterize both profit maximization and a heterogeneous cluster [13–16]. The authors of [13] devise an online algorithm, in which they provide a provable competitive ratio for the dynamic profit maximization problem in a Mobile Edge Computing (MEC) network. The scheme in [14] proposes an online workload scheduling strategy named OnSocMax to solve convex pseudo-social welfare maximization problems. Authors in [15] describe a novel framework that dynamically strives to ensure the successful completion of the maximum number of jobs before their deadlines within heterogeneous GPU clusters. The scheme in [16] categorizes heterogeneous EVs that follow similar charging profiles during dynamic charging. Thus, we propose a novel online scheduling model which can characterize both profit maximization and a heterogeneous cluster.

This paper proposes an online scheduling policy to maximize system profit under a discrete time-slot setting by all scheduling decisions executed at the start of each time interval. The proposed method models heterogeneous approximation jobs with distinct utility functions and runtime limits, while differentiating machines by their varied resource usage profiles. Our work resolves this optimization task using primal–dual techniques. First, this work relaxed the integer program into a linear program and constructs a pseudo-tree. This paper then defines an objective function to select qualified jobs from the pseudo-tree. With the dual-fitting analytical tool, this work splits the main optimization problem into two subproblems to examine per-step increments of primal and dual objectives. Moreover, this paper extends our initial convex cost model to a linear cost formulation and designs efficient scheduling algorithms for this extended scenario.

- We design a brand-new primal–dual online allocation framework for profit maximization scheduling. Traditional models fail to jointly model profit objectives and heterogeneous clusters.
- We select eligible jobs by maximizing the overall objective while respecting per-machine capacity constraints. Our online algorithm generates optimal matching results without internal loops by linear complexity $\mathcal{O}(N)$ with improved efficiency.
- We propose an innovative way to quantify step-wise primal and dual objective increments by supporting finer-grained dual-fitting analysis. The carefully designed dual variables advance state-of-the-art dual-fitting tools and produce drastically tighter competitive guarantees.

We first review prior research in Section 2. Section 3 thereafter defines the system architecture and mathematical problem statement. The competitive performance analysis and online algorithm design are provided in Section 4, while Section 5 specifies a linear server cost formulation. Numerical simulations are carried out in Section 6 to assess the performance of our proposed online selection algorithm, and this work wraps up the whole work with concluding remarks in Section 7 (Table 1).

Table 1. All notations in our model.

Notation	Corresponding Meaning
a_j	the arrival time of job j
d_j	the leaving time of job j
f_j	the utility of job j
s	the power of utility function
h_j	the constant value served to different utility functions
g_{it}	the cost of machine i at time t
m	the power of cost function
k_i	the constant value served to different cost functions
M	the number of machines
N	the number of jobs
$x_j^i(t)$	the decision of scheduling
C_i	the capacity of machine i
α_j	the dual variable in OSA model
β_i	the dual variable in OSA model
θ_i	the constant value for approximation algorithm
λ_j	the dual variable in OSA + L model
γ_i	the dual variable in OSA + L model
P	the finally primal value
P^t	each step of the primal value
D	the finally dual value
D^t	each step of the dual value
P^*	the optimal value
δ	the competitive ratio
r_i	the CPU resource consumption
$y_j(t)$	the total number of tasks completed
$z_i(t)$	the overall resource load

2. Related Work

Partial execution for approximation jobs is a fast-growing research area tackling fundamental bottlenecks in algorithm development, resource management and cluster performance tuning. Reference [17] builds a parallel processing assumption for all jobs. Xu et al. focus on utility maximization via strategic task selection in [4]. Min-cost resource allocation is a widely explored research stream covered in [18,19]. Specifically, You et al. in [18] allocate resources and services with minimal expenditure by adjusting bid values over time. The study in [19] constructs a decision framework to enable low-cost and

real-time job scheduling within computing infrastructures. Shrouf et al. put forward a mathematical optimization model in [20] by offering a decision paradigm with broad real-world applicability.

In [21,22], the primal–dual approach is utilized to design online algorithms for various problems. This approach has proven to be effective in addressing these problems and achieving desirable outcomes. In addition, recent studies have extended the primal–dual approach to deal with convex optimization problems [23,24]. These extensions demonstrate the versatility and applicability of the primal–dual approach in different optimization scenarios. The dual-fitting method, developed in [22], is primarily used as a tool for analyzing designed algorithms. Based upon their framework [25], Xu et al. utilize round-robin scheduling and dual-fitting techniques to devise an online algorithm with a competitive ratio.

In [13], they adopt batch quasi-static solving without rolling incremental online updates and its multi-resource extension lacks competitive ratio guarantees with no native KKT reconstruction tailored for profit maximization objectives. Ref. [26] implements fully decentralized peer-to-peer computation, while our current framework coordinates dual variables and can be extended in future work. Second, Ref. [27] embeds dedicated accuracy-loss tradeoff terms for ML pipelines, while our cost function iteration-free linear computation ensures deterministic low latency without hyperparameter tuning. Ref. [27] relies on ADMM iterations with per-slot complexity $\mathcal{O}(I \cdot N)$, where I is the number of convergence iterations. Ref. [26] uses nested alternating gradient iterations and yields higher complexity $\mathcal{O}(I_1 \cdot I_2 \cdot N)$. Our online algorithms derive optimal assignments via closed-form KKT dual solutions without any inner iteration, achieving strictly linear complexity $\mathcal{O}(N)$ per time slot.

3. System Model

Our system model adopts an M-node computing cluster, where machines are labeled $1, 2, \dots, M$. This model employs a slotted time model, such that online job j arrives at timestamp a_j . The set of job arrival instants $(a_1, a_2, \dots, a_N)$ forms an arbitrary deterministic sequence. Each job j has an expiration deadline d_j by which it must leave the cluster. All tasks are approximation jobs supporting partial computation. Each machine i can run several jobs in parallel per time slot without exceeding its capacity limit. Upon arrival, job j joins a global queue managed by a cluster. Furthermore, this job waits to be assigned to available machines for execution. Upon arrival, each job's release time, deadline, resource demand, and cost function are fully known to the scheduler. Jobs are preemptible and resumable with no preemption cost.

3.1. Utility Maximization

Following the settings established in the literature [3,4,28], every job is assigned an exclusive utility function. More specifically, job j has a concave utility function $f_j(\cdot)$ with respect to y_j , which is the total number of tasks completed by deadline d_j . The scheduling decision variable $x_j^i(t)$ represents whether job j is served by machine i in time slot t and $y_j = \sum_{t=a_j}^{d_j} \sum_{i=1}^M x_j^i(t)$. This model adopts the utility function form proposed in [28] within our model and the mathematical expression of $f_j(\cdot)$ is given by

$$f_j(y_j) = h_j \cdot (y_j)^s, \quad (1)$$

We enforce the bound $0 < s < 1$. All values h_j for $j = 1, 2, \dots, N$ are known by the cluster system and quantify job heterogeneity. This yields the utility maximization problem stated as follows:

$$\max_{\{x(t)\}} \sum_{j=1}^N f_j \left(\sum_{t=a_j}^{d_j} \sum_{i=1}^M x_j^i(t) \right) \quad (2a)$$

$$\text{s.t.} \quad a_j \leq t \leq d_j, \quad (2b)$$

$$x_j^i(t) \in \{0, 1\}, \quad \forall j, i, t. \quad (2c)$$

Here, the constraint (2b) means that job j is available in $[a_j, d_j]$, which is the job's processing window.

3.2. Cost Minimization

Consistent with existing models in [5,8,29], our setup incorporates heterogeneous machines with job-processing overheads. Costs accrue additively for every individual time slot. Each machine i is equipped with a convex cost function g_{it} dependent on $z_i(t)$, the overall resource load of machine i at slot t , where $z_i(t) = \sum_{j=1}^N x_j^i(t)$. Following the cost function structure from [8], this paper writes g_{it} as

$$g_{it}(z_i(t)) = k_i \cdot (z_i(t))^m, \quad (3)$$

We impose the condition $m > 1$. r_i stands for the CPU resource consumption. The sum of resource demands from all assigned tasks is bounded by each machine's capacity C_i . All parameters k_i ($i = 1, 2, \dots, M$) and r_i are observable by the cluster system. k_i distinguishes distinct machine characteristics in the cost formulation. This leads to the formal statement of the cost minimization problem:

$$\min_{\{x(t)\}} \sum_{i=1}^M \sum_{t=0}^{\infty} g_{it} \left(\sum_{j=1}^N x_j^i(t) \right) \quad (4a)$$

$$\text{s.t.} \quad \sum_{j=1}^N x_j^i(t) \cdot r_i \leq C_i, \quad \forall i, \quad (4b)$$

$$a_j \leq t \leq d_j, \quad (4c)$$

$$x_j^i(t) \in \{0, 1\}, \quad \forall j, i, t. \quad (4d)$$

3.3. Profit Maximization Optimization Problem

This model introduces two vectors for scheduling decisions: $x_j(t) = \{x_j^1(t), x_j^2(t), \dots, x_j^M(t)\}$ for individual job j , and the aggregated vector $x(t) = \{x_1(t), x_2(t), \dots, x_N(t)\}$. The corresponding profit maximization problem reads

$$\max_{\{x(t)\}} \sum_{j=1}^N f_j \left(\sum_{t=a_j}^{d_j} y_j(t) \right) - \sum_{i=1}^M \sum_{t=0}^{\infty} g_{it}(z_i(t)) \quad (P1)$$

$$\text{s.t.} \quad y_j(t) = \sum_{i=1}^M x_j^i(t),$$

$$z_i(t) = \sum_{j=1}^N x_j^i(t),$$

$$x_j^i(t) = 0, t \notin [a_j, d_j].$$

Equations (2b), (2c), (4b) are satisfied.

4. The Dual-Fitting Framework

In Problem (P1), profit computation is only feasible after jobs complete processing and exit the cluster. As a result, direct online resolution of this optimization model is

infeasible. This work addresses Problem (P1) via the primal–dual framework, noting its first two constraints are linear. We assign Lagrangian dual vectors $\alpha = (\alpha_j, j = 1, \dots, N)$ and $\beta = (\beta_i, i = 1, \dots, M)$ to the first and second constraints separately. This transforms Problem (P1) into the following equivalent formulation:

$$\begin{aligned} \min_{\alpha \geq 0, \beta \geq 0} \max_p \quad & \Phi(p, \alpha, \beta) \\ \text{s.t.} \quad & x_j^i(t) \in \{0, 1\}, \end{aligned} \quad (\text{D1})$$

where $\Phi(p, \alpha, \beta)$ can be calculated by

$$\begin{aligned} \Phi(p, \alpha, \beta) = & \sum_{j=1}^N f_j \left(\sum_{t=a_j}^{d_j} y_j(t) \right) - \sum_{i=1}^M \sum_{t=0}^{\infty} g_{it}(z_i(t)) \\ & - \sum_{j=1}^N \sum_{t=a_j}^{d_j} \alpha_j(t) (y_j(t) - \sum_{i=1}^M x_j^i(t)) \\ & + \sum_{i=1}^M \sum_{t=a_j}^{d_j} \beta_i(t) (z_i(t) - \sum_{j=1}^N x_j^i(t)) \end{aligned} \quad (5)$$

The dual problem (D1) can be regarded as two optimization problems A_1 and B_1 . This model can use A_1 to design an approximation algorithm and use B_1 to calculate dual variables. Thus, we can get

$$\begin{aligned} \min_{\alpha, \beta} \quad & A_1 + B_1 \\ \text{where} \quad & A_1 = \max \left\{ \sum_{j=1}^N \sum_{i=1}^M \sum_{t=a_j}^{d_j} x_j^i(t) (\alpha_j(t) - \beta_i(t)) \right\} \\ \text{s.t.} \quad & \alpha_j(t) \geq 0, \quad \forall j, t, \quad \beta_i(t) \geq 0, \quad \forall i, t, \\ & A_1 \geq 0, \quad \forall i, j, t, \\ & x_j^i(t) \in \{0, 1\}, \quad \forall j, i, \end{aligned} \quad (\text{D2})$$

Equations (2b), (2c), (4b) are satisfied.

B_1 is given by

$$\begin{aligned} B_1 = \max_{\alpha \geq 0, \beta \geq 0} \{ & \sum_{j=1}^N f_j \left(\sum_{t=a_j}^{d_j} y_j(t) \right) - \sum_{i=1}^M \sum_{t=0}^{\infty} g_{it}(z_i(t)) \\ & - \left(\sum_{j=1}^N \sum_{t=a_j}^{d_j} \alpha_j(t) y_j(t) - \sum_{i=1}^M \sum_{t=a_j}^{d_j} \beta_i(t) z_i(t) \right) \} \end{aligned} \quad (6)$$

The Lagrangian decomposes additively into job-only terms and node-only terms with no cross-coupling, which mathematically justifies why independent subproblems (A_1) and (B_1) can be solved separately. The job utility function can be reformulated by summing multiple timely dependent functions as

Lemma 1. *There is a function sequence $T_{jt}(y_j(t))$ such that $f_j(\sum_{t=a_j}^{d_j} y_j(t)) = \sum_{t=a_j}^{d_j} T_{jt}(y_j(t))$.*

We prove Lemma 1 in Appendix A.1. With Lemma 1, this work can get a sub-optimization problem in each time slot.

4.1. Online Scheduling Algorithm Design

From the above analysis, subproblem A_1 can be solved slot by slot, which delivers an online job allocation strategy and removes summations over multiple time intervals. The values of $\alpha_j(t)$ and $\beta_i(t)$ are computed based on the assignment variables $x_j^i(t-1)$ from the prior slot. We enforce two conditions on $x_j^i(t)$: binary integrality $x_j^i(t) \in \{0, 1\}$ and the machine capacity bound $\sum_{j=1}^N x_j^i(t) r_i \leq C_i$. These constraints together form the integer optimization problem (D3):

$$\max_{\{x(t)\}} \sum_{j=1}^N \sum_{i=1}^M x_j^i(t) \cdot (\alpha_j(t) - \beta_i(t)) \quad (\text{D3})$$

$$\text{s.t.} \quad \alpha_j(t) \geq 0, \quad \forall j, t, \quad \beta_i(t) \geq 0, \quad \forall i, t, \\ \alpha_j(t) > \beta_i(t) \quad \forall j, i, t,$$

Equations (2b), (2c), (4b) are satisfied.

This work relaxed the binary constraints of $x_j^i(t)$ to the continuous range $[0, 1]$ for solving Problem (D3), where the total sum of all relaxed variables is normalized to one. This paper then picks the entry with the maximum fractional value and rounds it into an integer assignment. Referring to [30], we construct a pseudo-tree composed of jobs and machines, which is visualized in Figure 1. The pseudo-tree defines layered message passing paths between nodes and the central coordinator. Local dual residuals only propagate upward along parent–child edges within the tree, avoiding full mesh peer-to-peer transmission to cut communication overhead. It is built in three fixed tiers with static unchanging topology initialized before scheduling begins. Each node preallocates dedicated storage for global/local dual multipliers. Job progress and residual statistics bound to parent–child quantized communication edges. This paper models all jobs as leaf nodes and each machine as a root node. We add an edge between machine i and job j if $x_j^i(t) > 0$. The algorithm picks the optimal child job j of machine i by maximizing $x_j^i(t)(\alpha_j(t) - \beta_i(t))$ over all $j \in \{1, 2, \dots, N\}$, and validates the capacity bound $\sum_{j=1}^N x_j^i(t) r_i \leq C_i$. When the capacity condition is met, one task of job j is allocated to machine i with $x_j^i(t)$ set to one, and job j is removed from the tree in Figure 1. If the constraint is breached, the job gets rejected and its connected edge is removed. Rejected jobs are reallocated to other machines, generating a new pseudo-tree for slot t . The algorithm continues searching for the next capacity-compliant child job for machine i . Updates proceed in a fixed three-stage cycle. Leaves run local KKT allocation and upload updated deadline duals bottom-up. Intermediates aggregate subtree data for relay. The root computes refreshed global dual variables via dual-fitting projection and broadcasts them top-down to all leaves. This prohibits splitting one job across multiple machines in the same slot. For tractable convex optimization, we relax $x_j^i(t) \in \{0, 1\}$ to continuous domain $x_j^i(t) \in [0, 1]$. After solving the relaxed convex program, a dedicated rounding algorithm maps continuous fractional solutions back to feasible binary assignments while preserving the above single-node allocation constraint.

For the dual variables in the next time slot, this paper uses the same method in [8] which adopts the primal–dual approach ($\theta_i > 1$). After the time slot t , this work can get the dual variables $(\alpha_j(t+1)$ and $\beta_i(t+1))$ based on the current assignment $(x_j^i(t))$ as follows:

$$\alpha_j(t+1) = T'_{jt}(y_j(t)) = \frac{dT_{jt}(y_j(t))}{d(y_j(t))}. \quad (7)$$

$$\beta_i(t+1) = g'_{it}(\theta_i z_i(t)) = \frac{dg_{it}(\theta_i z_i(t))}{d(\theta_i z_i(t))}. \quad (8)$$

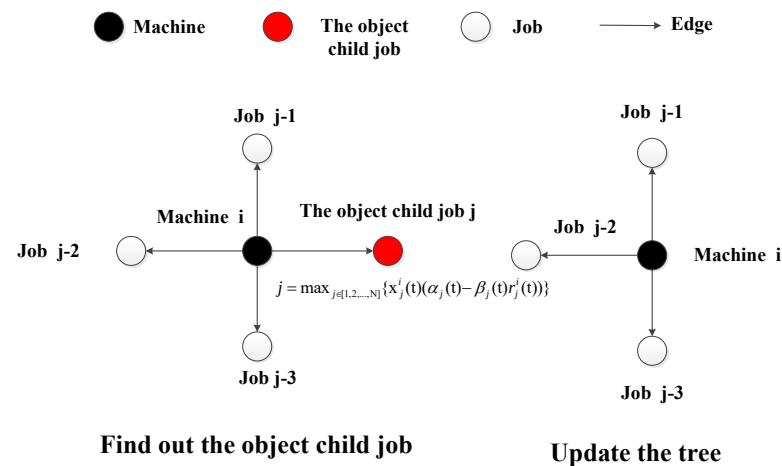


Figure 1. The pseudo-tree in our model.

For time slot $t + 1$, the online algorithm computes the subsequent assignment based on dual variables. Algorithm 1 provides the pseudo-code for the online scheduling algorithm (OSA). As justified by the KKT equations, the OSA achieves a worst-case time complexity of $\mathcal{O}(N)$ [31]. We formally derive per-slot complexity as $\mathcal{O}(N)$. Each of the N jobs undergoes constant-cost closed-form KKT allocation with no inner iterations. Intermediate aggregation sums leaf outputs at total cost bounded by $\mathcal{O}(N)$, while root dual updates take constant $\mathcal{O}(1)$ time independent of job count. For D resource dimensions complexity extends to $\mathcal{O}(D \cdot N)$ with fixed small D . This linear polynomial complexity guarantees high scalability and low-latency updates for real-time large-scale scheduling scenarios. Our algorithm outperforms iterative optimization schedulers $\mathcal{O}(I \cdot N)$ [32] and sorting-based greedy algorithms $\mathcal{O}(N \log N)$ [33] in the existing literature, delivering faster per-slot computation for real-world large-scale online scheduling. These full-information preemptive assumptions underpin our KKT-based closed-form online update and directly determine the achievable competitive ratio of the proposed algorithm. The auxiliary term T_{jt} is formally defined as the subgradient evaluated at online-observable cumulative job progress. We explicitly verify that T_{jt} can be computed at slot t using only historical state and carries no dependence on future allocation decisions. The algorithm avoids naive $\mathcal{O}(MN)$ exhaustive job-machine matching via the single-machine job assignment constraint and layered pseudo-tree aggregation, with no nested (j, i) pairwise loops. Under realistic cloud settings with static fixed hardware pool M and constant small D , the dominant workload-dependent term yields linear $\mathcal{O}(N)$ scaling for online scheduling.

4.2. Competitive Performance Analysis

In this section, we use a dual-fitting method to analyze the competitive performance of the OSA. Let P^t denote the primal objective value in time slot t which is considered as the step t . D^t denotes the dual objective value in step t . P is the total primal objective value of all steps in the (P1) problem and D is the total dual objective value of all steps in the (D1) problem. Based on the above definition, we get to know the initial values $P^0 = 0$ and $D^0 = 0$. The increase of two steps for the primal objective value is $P^t - P^{t-1}$ and the increase of two steps for the dual objective value is $D^t - D^{t-1}$. Thus, P^T and D^T can be shown as following:

$$P^T = \sum_t (P^t - P^{t-1}). \quad (9)$$

$$D^T = \sum_t (D^t - D^{t-1}). \quad (10)$$

Algorithm 1 Online Scheduling Algorithm with the Convex Cost Function

```

1: Input:  $r_i, \alpha_j(t-1), \beta_i(t-1)$ ;
2: Output:  $x_j^i(t), \alpha_j(t), \beta_i(t)$ ;
3: Initialize :  $x_j^i(t) = 0$  and calculate  $\alpha_j(t), \beta_i(t)$ ;
4: When jobs and machines arrive at the system, the system constructs a pseudo-tree in
   time slot  $t$ ;
5: Get the dual variables  $\alpha_j(t) = \alpha_j(t-1)$  and  $\beta_i(t) = \beta_i(t-1)$ ;
6: while jobs are available do
7:   Find a child job  $j$  for machine  $i$  which is satisfied:
8:    $j = \max_{j \in [1, 2, \dots, N]} \{x_j^i(t)(\alpha_j(t) - \beta_i(t) \cdot r_i)\}$ ;
9:   if  $\sum_{j=1}^N x_j^i(t)r_i \leq C_i$ ; then
10:    Allocate job  $j$  to machine  $i$ ;
11:     $x_j^i(t) = 1$ ;
12:    Calculate the utility and the costs;
13:    Update the dual variables  $\alpha_j(t)$  and  $\beta_i(t)$ 
14:    based on  $T'_{jt}(y_j(t))$  and  $g'_{it}(\theta_i z_i(t))$ ;
15:    Remove job  $j$  from the constructed tree;
16:   else
17:     $x_j^i(t) = 0$ , Reject job  $j$  for machine  $i$ ;
18:   end if
19: end while

```

Lemma 2. *If there is a constant $\delta_1 \geq 1$ such that the increase in the primal objective value and the increase in the dual objective value differ by at most a δ_1 factor, i.e., $P^t - P^{t-1} \geq \frac{1}{\delta_1}(D^t - D^{t-1})$, then, this work can get that the algorithm is δ_1 competitive.*

Proof. Sum up $P^t - P^{t-1}$ and $D^t - D^{t-1}$ for all steps; it can be seen that

$$\sum_t (P^t - P^{t-1}) = (P^t - P^0). \quad (11)$$

$$\sum_t (D^t - D^{t-1}) = (D^t - D^0). \quad (12)$$

Based on $P^t - P^{t-1} \geq \frac{1}{\delta_1}(D^t - D^{t-1})$, it follows that

$$\sum_t (P^t - P^{t-1}) \geq \frac{1}{\delta_1} \sum_t (D^t - D^{t-1}). \quad (13)$$

Thus, it can be seen that

$$(P^t - P^0) \geq \frac{1}{\delta_1}(D^t - D^0). \quad (14)$$

Here, it can be seen that $P^0 = 0$, $D^0 = 0$, $\sum_t P^t = P$ and $\sum_t D^t = D$. Thus, we have $D \leq \delta_1 \cdot P$. By weak duality [34], $P \leq P^* \leq D$. The optimal solutions can produce optimal value P^* for all steps. Therefore, $P^* \leq D \leq \delta_1 \cdot P$ and the algorithm is δ_1 competitive. $\square$

Lemma 3. *The maximum value of $B_1(\alpha_j(t), \beta_i(t))$ for job j on machine i in time slot t is*

$$B_1^*(\beta_i(t)) = (1 - \frac{1}{m})(\frac{1}{k_i m})^{\frac{1}{m-1}} \beta_i(t)^{\frac{m}{m-1}}. \quad (15)$$

where $z_i^*(t) = (\frac{\beta_i(t)}{k_i m})^{\frac{1}{m-1}}$. We prove Lemma 3 in Appendix A.2.

To get δ_1 in Lemma 2, this study shows that the dual variables $\alpha_j(t)$ and $\beta_i(t)$ satisfy an Allocation–Profit Relationship, which can guarantee the inequality in Lemma 2.

Definition 1. For a given parameter $\delta_j^i \geq 1$, the Differential Allocation–Profit Relationship is

$$\beta_i(t)dz_i(t) - g'_{it}(z_i(t))dz_i(t) \geq \frac{1}{\delta_j^i} [B_1^{*}(\beta_i(t))d\beta_i(t)]. \quad (16)$$

Lemma 4. For $m > 1$, δ_j^i is

$$\delta_j^i = 4m. \quad (17)$$

Lemma 4 is proved in Appendix A.3.

Theorem 1. For $\delta_j^i \geq 1$, it can be seen that $\delta_1 = \delta_j^i = 4m$.

Proof. Consider Lemma 1 and the primal problem (P1); the increase in the primal objective value $P^t - P^{t-1}$ is given by

$$P^t - P^{t-1} = T_{jt}(y_j(t)) - T_{jt-1}(y_j(t-1)) - (g_{it}(z_i(t)) - g_{it-1}(z_i(t-1))) \quad (18)$$

Since $f_j(\sum_{t=a_j}^{d_j} y_j(t))$ is a concave function, by the Taylor approximation, this work can get

$$T_{jt}(y_j(t)) - T_{jt-1}(y_j(t-1)) \geq T'_{jt}(y_j(t))(y_j(t) - y_j(t-1)). \quad (19)$$

Moreover, $g_{it}(\sum_{t=a_j}^{d_j} z_i(t))$ is a convex function. By the Taylor approximation, this paper also can get

$$g_{it}(z_i(t)) - g_{it-1}(z_i(t-1)) \leq g'_{it}(z_i(t))(z_i(t) - z_i(t-1)). \quad (20)$$

Let $y_j(t) - y_j(t-1) = dy_j(t)$ and $z_i(t) - z_i(t-1) = dz_i(t)$. $dy_j(t)$ is considered as the increase of $y_j(t)$ in time slot t and $dz_i(t)$ is considered as the increase of $z_i(t)$ in slot t . Thus, it can be seen that

$$P^t - P^{t-1} \geq T'_{jt}(y_j(t))dy_j(t) - g'_{it}(z_i(t))dz_i(t) \quad (21)$$

Based on Equations (7) and (8), we can get $d\alpha_j(t) \leq d\beta_i(t)$. Thus, $A_1^t - A_1^{t-1}$ can be further satisfied with

$$A_1^t - A_1^{t-1} \leq \alpha_j(t)dy_j(t) - \beta_i(t)dz_i(t) \quad (22)$$

From the online scheduling algorithm, we can know that one task of job j is given to machine i in each time slot t . Thus, $T'_{jt}(y_j(t)) = T'_{jt-1}(y_j(t-1))$. By $T'_{jt-1}(y_j(t-1)) = \alpha_j(t)$ in Equation (7), it follows that

$$P^t - P^{t-1} - (A_1^t - A_1^{t-1}) \geq \beta_i(t)dz_i(t) - g'_{it}(z_i(t))dz_i(t). \quad (23)$$

Note that $\beta_i(t)$ is the variable in B_1^* , which is a convex function. Based on Equation (15), we can show $B_1^{*t} - B_1^{*(t-1)}$ as follows:

$$\begin{aligned} B_1^{*t} - B_1^{*(t-1)} &= \left(1 - \frac{1}{m}\right) \left(\frac{1}{k_i m}\right)^{\frac{1}{m-1}} \beta_i(t)^{\frac{m}{m-1}} - \left(1 - \frac{1}{m}\right) \left(\frac{1}{k_i m}\right)^{\frac{1}{m-1}} \beta_i(t-1)^{\frac{m}{m-1}} \\ &= \left[\beta_i(t) \left(1 - \frac{1}{m}\right) \left(\frac{\beta_i(t)}{k_i m}\right)^{\frac{1}{m-1}} \right]' d\beta_i(t) \\ &\leq B_1'^*(\beta_i(t)) d\beta_i(t) \end{aligned} \quad (24)$$

Based on $\alpha_j(t)y_j(t) \geq \beta_i(t)z_i(t)$ and Lemma 2, we can get $A_1^t - A_1^{t-1} \geq 0$. By $\delta_j^i \geq 1$, it can be seen that

$$P^t - P^{t-1} \geq \frac{1}{\delta_j^i} [D^t - D^{t-1}]. \quad (25)$$

□

Based on Lemma 2, it can be seen that

$$\delta_1 = \delta_j^i = 4m. \quad (26)$$

where $m > 1$. The OSA algorithm is a $4m$ -competitive.

Our dual-fitting workflow updates dual variables per time slot based on constraint residuals generated from the KKT primal online scheduling decision, with projection operations to maintain permanent dual feasibility. The closed-form KKT solution naturally guarantees primal feasibility for resource, deadline and quantized channel constraints. Using weak primal–dual duality, we derive a complete chain of inequalities to obtain a constant competitive ratio bound for the OSA.

5. Case Study: Linear Server Costs

In this part, we use our model to handle the linear cost function, which describes another analysis method. To be more specific, this paper let $m = 1$ for Equation (3) in Section 3.2. Thus, the linear cost function can be reformulated as follows:

$$g_{it} \left(\sum_{j=1}^N x_j^i(t) \cdot r_i \right) = k_i \cdot \left(\sum_{j=1}^N x_j^i(t) \cdot r_i \right), \quad (27)$$

Similarly, with the linear cost function, our new optimization model can be given by

$$\begin{aligned} \max_{\{x(t)\}} \quad & \sum_{j=1}^N f_j \left(\sum_{t=a_j}^{d_j} y_j(t) \right) - \sum_{i=1}^M \sum_{t=a_i}^{d_i} k_i \cdot z_i(t) \\ \text{s.t.} \quad & y_j(t) = \sum_{i=1}^M x_j^i(t), \\ & z_i(t) = \sum_{j=1}^N x_j^i(t) \cdot r_i, \\ & x_j^i(t) = 0, t \notin [a_j, d_j], \end{aligned} \quad (P2)$$

Equations (2b), (2c), (4b) are satisfied.

The primal problem (P2) cannot be efficiently solved online via direct methods due to its inherent intractability. This model overcomes difficulty by applying the primal–dual framework to handle the optimization problem. It can be observed that both the

first and second constraints of (P2) belong to linear constraints. We assign Lagrangian dual vectors λ and γ to the two groups of constraints, where $\lambda = (\lambda_j, j = 1, \dots, N)$ and $\gamma = (\gamma_i, i = 1, \dots, M)$. Accordingly, the transformed formulation of Problem (P2) reads

$$\begin{aligned} \min_{\lambda \geq 0, \gamma \geq 0} \max_p \quad & \Phi(p, \lambda, \gamma) \\ \text{s.t.} \quad & x_j^i(t) \in \{0, 1\}, \end{aligned} \quad (\text{D1})$$

where $\Phi(p, \lambda, \gamma)$ is given by

$$\begin{aligned} \Phi(p, \lambda, \gamma) = & \sum_{j=1}^N f_j \left(\sum_{t=a_j}^{d_j} y_j(t) \right) - \sum_{i=1}^M \sum_{t=a_j}^{d_j} g_{it} \left(z_i(t) \right) \\ & - \sum_{j=1}^N \sum_{t=a_j}^{d_j} \lambda_j(t) \left(y_j(t) - \sum_{i=1}^M x_j^i(t) \right) \\ & + \sum_{i=1}^M \sum_{t=a_j}^{d_j} \gamma_i(t) \left(z_i(t) - \sum_{j=1}^N x_j^i(t) \cdot r_i \right) \end{aligned} \quad (\text{28})$$

This decomposes the dual problem (D1) into two subproblems: A_2 yields an approximation algorithm and B_2 calculates dual variables from the job assignment. These lead to the following optimization problems:

$$\begin{aligned} \min_{\lambda, \gamma} \quad & A_2 + B_2 \\ \text{where} \quad & A_2 = \max \left\{ \sum_{j=1}^N \sum_{i=1}^M \sum_{t=a_j}^{d_j} x_j^i(t) \left(\lambda_j(t) - \gamma_i(t) \cdot r_i \right) \right\}, \\ \text{s.t.} \quad & \lambda_j(t) \geq 0, \quad \forall j, t, \quad \gamma_i(t) \geq 0, \quad \forall i, t, \\ & A_2 \geq 0, \quad \forall i, j, t, \\ & \text{Equations (2b), (2c), (4b) are satisfied.} \end{aligned} \quad (\text{D2})$$

B_2 is given by

$$\begin{aligned} B_2 = & \max_{\lambda \geq 0, \gamma \geq 0} \left\{ \sum_{j=1}^N f_j \left(\sum_{t=a_j}^{d_j} y_j(t) \right) - \sum_{i=1}^M \sum_{t=a_j}^{d_j} g_{it} \left(z_i(t) \right) \right. \\ & \left. - \left(\sum_{j=1}^N \sum_{t=a_j}^{d_j} \lambda_j(t) \cdot y_j(t) - \sum_{i=1}^M \sum_{t=a_j}^{d_j} \gamma_i(t) \cdot z_i(t) \right) \right\} \end{aligned} \quad (\text{29})$$

5.1. Algorithm Design

In this part, the dual variables will be updated by every job assignment in slot t . Moreover, we calculate the current $\lambda_j(t)$ and $\gamma_i(t)$ as follows:

$$\lambda_j(t) = T'_{jt}(y_j(t)), \gamma_i(t) = k_i. \quad (\text{30})$$

where $\lambda_j(t) \sum_{i=1}^M x_j^i(t) \geq k_i \sum_{j=1}^N x_j^i(t) r_i$.

For time slot $t + 1$, the online algorithm derives new job allocation results through dual variables. The complete pseudocode of OSA + L (online scheduling algorithm with linear cost function) is illustrated in Algorithm 2. According to the KKT equations and the theoretical scheme proposed in [31], OSA + L features a worst-case computational

complexity no greater than $\mathcal{O}(N)$. Similarly, such linear polynomial complexity enables stable scalable execution under massive real-time task loads.

Algorithm 2 Online Scheduling Algorithm with the Linear Cost Function

```

1: Input:  $r_i, \lambda_j(t-1), \gamma_i(t-1)$ ;
2: Output:  $x_j^i(t), \lambda_j(t), \gamma_i(t)$ ;
3: Initialize :  $x_j^i(t) = 0$  and calculate  $\lambda_j(t), \gamma_i(t)$ ;
4: When jobs and machines arrive at the system, the system constructs a pseudo-tree in
   time slot  $t$ ;
5: Get the dual variables  $\lambda_j(t) = \lambda_j(t-1)$  and  $\gamma_i(t) = \gamma_i(t-1)$ ;
6: while jobs are available do
7:   Find a child job  $j$  for machine  $i$  which is satisfied:
8:    $j = \max_{j \in [1, 2, \dots, N]} \{ \lambda_j(t)x_j^i(t) - k_i x_j^i(t)r_i \}$ ;
9:   if  $\sum_{j=1}^N x_j^i(t) \leq C_i$  then
10:    Allocate job  $j$  to machine  $i$ ;
11:     $x_j^i(t) = 1$ ;
12:    Calculate the utility and the costs;
13:    Update  $\lambda_j(t)$  and  $\gamma_i(t)$ 
14:    based on  $T'_{jt}(y_j(t))$  and  $k_i$ ;
15:    Remove job  $j$  from the constructed tree;
16:   else
17:     $x_j^i(t) = 0$ , Reject job  $j$  for machine  $i$ 
18:   end if
19: end while

```

5.2. Performance of OSA + L

For problem (P2), P^t can be given by

$$P^t = T_{jt}(y_j(t)) - k_i \cdot z_i(t) \quad (31)$$

In this dual problem, D^t can be expressed by

$$\begin{aligned}
 D^t &= A_2^t + B_2^t \\
 &= \lambda_j(t) \cdot \sum_{i=1}^M x_j^i(t) - \gamma_i(t) \cdot \sum_{j=1}^N x_j^i(t)r_i + T_{jt}(y_j(t)) - k_i \cdot z_i(t) \\
 &\quad - (\lambda_j(t) \cdot y_j(t) - \gamma_i(t) \cdot z_i(t))
 \end{aligned} \quad (32)$$

With Lemma 1 $f'_j(\epsilon_j(t_n)) \cdot y_j(t_n) = T_{jt}(y_j(t_n))$. Based on Equation (7), we get $\lambda_j(t) = T'_{jt}(y_j(t)) = f'_j(\epsilon_j(t))$. It can be seen that

$$\lambda_j(t) \cdot y_j(t) = f'_j(\epsilon_j(t_n)) \cdot y_j(t) = T_{jt}(y_j(t)). \quad (33)$$

Inferred from Equation (30), it follows that

$$\begin{aligned}
 A_2^t &= \lambda_j(t) \cdot \sum_{i=1}^M x_j^i(t) - \gamma_i(t) \cdot \sum_{j=1}^N x_j^i(t)r_i \\
 &= \lambda_j(t) \cdot y_j(t) - \gamma_i(t) \cdot z_i(t) \\
 &= T_{jt}(y_j(t)) - k_i \cdot z_i(t) \\
 &= P^t
 \end{aligned} \quad (34)$$

By considering the constraint $\lambda_j(t) \sum_{i=1}^M x_j^i(t) \geq k_i \sum_{j=1}^N x_j^i(t) r_i$, it follows that

$$\begin{aligned} B_2^t &= T_{jt}(y_j(t)) - k_i \cdot z_i(t) - (\lambda_j(t) \cdot y_j(t) - \gamma_i(t) \cdot z_i(t)) \\ &\leq P^t \end{aligned} \quad (35)$$

Combining Equations (34) and (35), it can be seen that

$$D^t = A_2^t + B_2^t \leq 2 \cdot P^t \quad (36)$$

As we know $P = \sum_t P^t$ and $D = \sum_t D^t$. Thus, this paper can make a conclusion that $P \leq P^* \leq D \leq 2 \cdot P$ and $\delta_2 = 2$. The OSA + L algorithm is a two-competitive.

5.3. The Comparison Between δ_1 and δ_2

In these two models, we can get

$$\delta_1 = 4m, \delta_2 = 2 \quad (37)$$

Comparing δ_1 and δ_2 , this model can get $\delta_1 > \delta_2$ by $m > 1$. We conclude that the performance bound of the linear cost function model is better than the convex cost function model. The framework supports arbitrary D -dimensional heterogeneous resources. Per-slot complexity scales as $\mathcal{O}(D \cdot N)$ linear in dimension D . Unlike classic multi-dimensional scheduling approaches that suffer exponentially worsening competitive ratios with higher D , our decoupled dimension-wise dual-fitting updates yield a competitive bound scaling only linearly with D by avoiding explosive performance degradation for multi-resource joint scheduling. Our algorithm supports general concave marginal utility for partial outputs. Partial execution maps naturally to batch data processing systems, and indivisible jobs are a special constrained case of this framework. Cumulative progress values directly feed into the residual calculations driving our per-slot dual variable updates.

6. Performance Evaluation

This paper tests the performance of the online scheduler with Google cluster-usage traces [35]. The trace data provides each job j with its arrival time a_j and deadline d_j . This work imports all job and machine trace records into our cluster simulation environment. Consistent with [28], job utility is captured via the concave function $f_j(y) = h_j y^t$, where we set $t = 1/2$ and draw h_j randomly from $[1, 5]$. The cost model follows the convex function $g_{it}(z) = k_i z^m$ from [8]. For the OSA, this paper takes $m = 2$ and each k_i is assigned a random value between one and five. We deploy the online algorithm in a local private cluster of two two-socket physical hosts. Each host is configured with 64 CPU cores and 128 GB RAM. (Table 2).

Table 2. Google traces.

the number of jobs	6064
the duration (s)	35,032
the average number of tasks for job	26.31
the minimum duration of tasks	12.8
the maximum duration of tasks	22,919.3
the average duration of tasks	1179.7

This work provides full formal citations for the trace’s original publication and official schema documentation in the reference list, and include this explicit license statement as required for derivative research. This paper eliminates all evicted, failed and incomplete tasks lacking valid resource demand records. This work also discards outlier tasks whose resource requirements exceed the 99th percentile of the global workload distribution to avoid biased experimental results caused by extreme resource requests. This paper fixed all random seeds, resource scaling factors and trace segmentation ratios for full repeatability. We run 25 repeated trials for all test cases, report mean $\pm$ standard deviation with 95% confidence intervals, and mark statistically significant improvements via pairwise t -tests at $\alpha = 0.05$.

Baseline Schemes: The following three schemes are used as baselines to compare with our proposed algorithms:

- **LCF** (Lowest Cost First). The cluster scheduler selects jobs with the lowest consumption to schedule in each time slot, which is a classic greedy online allocation heuristic widely adopted.
- **EVC** (Electric Vehicle Charging). The cluster scheduler make decisions based on [29] in each time slot, which enables direct comparison between auction mechanisms and our rolling closed-form dual-fitting pipeline.
- **PDA** (Primal–Dual Online Algorithm). The cluster scheduler chooses jobs based on [8] in each time slot, which allows us to clearly demonstrate the complexity and latency advantages of our iteration-free $\mathcal{O}(N)$ closed-form solution.

We explicitly document unified adaptation rules applied to all baselines: identical online information access, fully consistent constraint sets, shared relaxation/rounding pipelines, and standardized hyperparameter tuning to guarantee fair comparison.

6.1. Performance of the OSA

Figure 2 reveals that the total profit reaches its maximum at a machine count of 1000, surpassing other test cases. Below 1000 machines, resource congestion creates long job queues, reduces completion speed and cuts total accumulated profit. From Figure 3, we observe that a 5 s scheduling interval delivers superior profit relative to other time intervals. At a 5 s window length, we strike a balance: the time granularity is fine enough to respond timely to dynamic burst job arrivals and avoid large residual progress backlogs, while the frequency of scheduling recalculation remains low enough to limit redundant computational overhead. We therefore adopt 1000 machines and a 5 s scheduling window as our default experimental setup. A reasonable machine quantity paired with shorter scheduling intervals enables the OSA to make refined allocation judgments and match jobs to proper machines precisely. As shown in Figure 4, the OSA algorithm reduces average job flowtime by more than 25%.

With the fixed parameters outlined above, we compare the OSA against three baseline algorithms (LCF, PDA, EVC) on the cluster testbed. Figure 5 tracks evolving total job utility across algorithms over time. The OSA’s utility curve is uniformly higher than all baselines. LCF and PDA produce tightly aligned utility values throughout simulation. EVC outperforms both LCF and PDA in utility output. Figure 5 quantifies a 20% utility gain brought by the OSA, as the primal–dual design of the OSA optimizes total utility at each time slot.

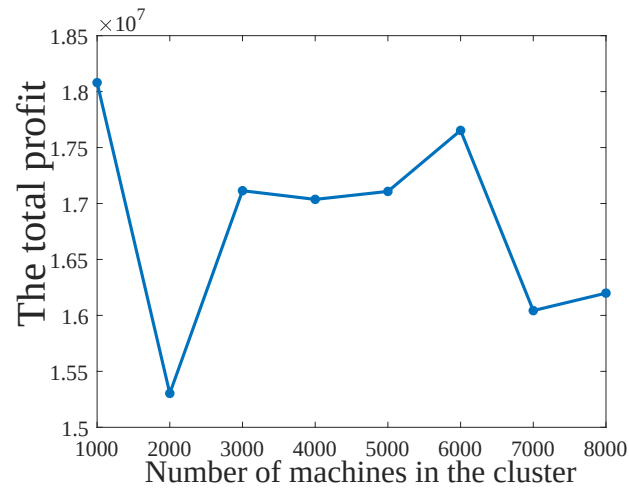


Figure 2. The total profit by the OSA.

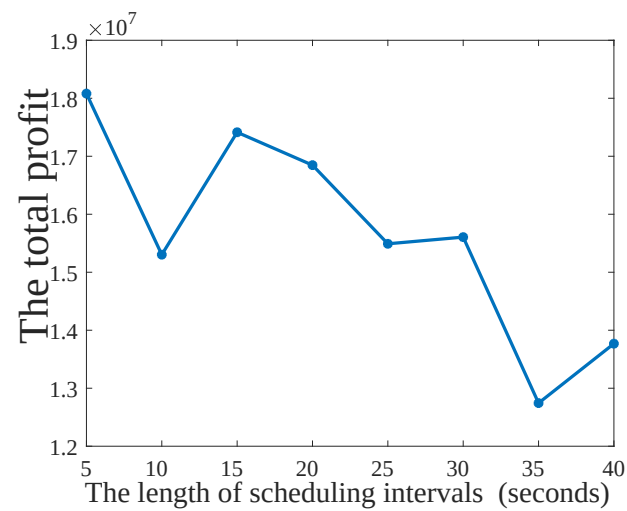


Figure 3. The total profit by the OSA.

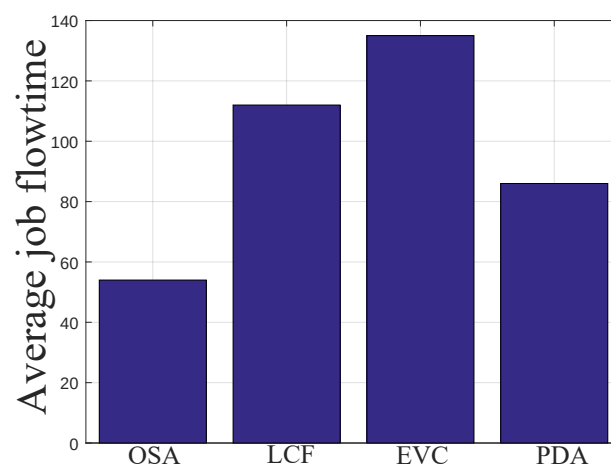


Figure 4. Different job flowtime.

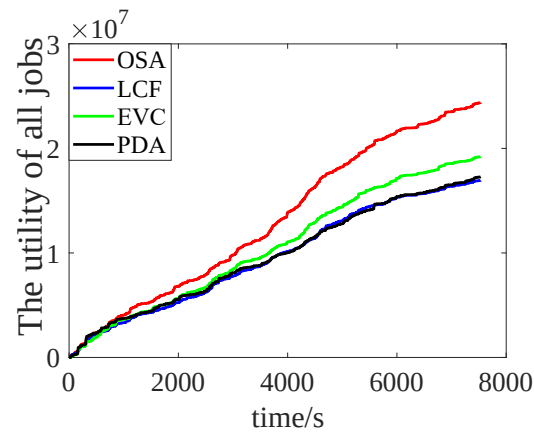


Figure 5. The total utility by the OSA.

Figure 6 presents dynamic machine cost profiles for each algorithm. The OSA maintains lower cost values than LCF, PDA and EVC at all timestamps, meaning the OSA achieves reduced aggregate overhead. The figure confirms nearly 15% cost reduction in the OSA versus baselines, attributed to the online primal–dual mechanism that minimizes instantaneous machine costs under per-slot assignments.

Time-varying total profit curves are plotted in Figure 7. The OSA consistently attains higher profit than all baselines and PDA exhibits the weakest profit performance among the four. This substantiates the OSA as an effective scheduler for profit-maximizing cluster workloads. The figure records a 35% improvement in practical total profit under the OSA compared with the three baselines, proving the competitive advantages of our proposed online algorithm.

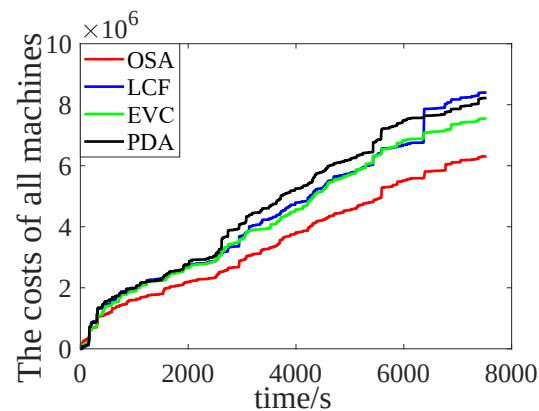


Figure 6. The total costs by the OSA.

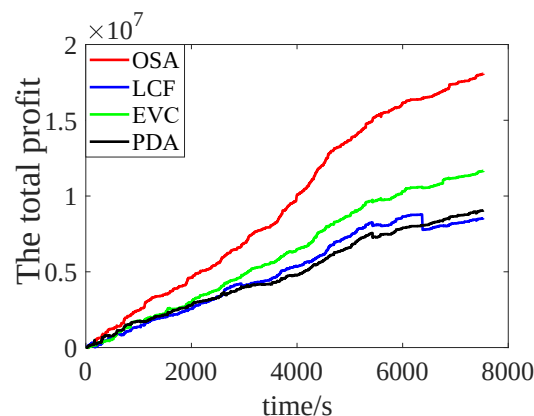


Figure 7. The total profit by the OSA.

6.2. Performance of OSA + L

This part also shows the performance of the online algorithm OSA + L by Google cluster-usage traces [35]. This chooses the number of machines to be 1K and a scheduling interval of 5 s for OSA + L. Figure 8 shows that OSA + L improves the total utility by 25% compared to the other three algorithms. In addition, Figure 9 presents that the total cost vector of OSA + L is close to the other three algorithms after all jobs are completed. Finally, Figure 10 shows that OSA + L can increase the performance of profit by nearly 30%, which implies OSA + L is more effective.

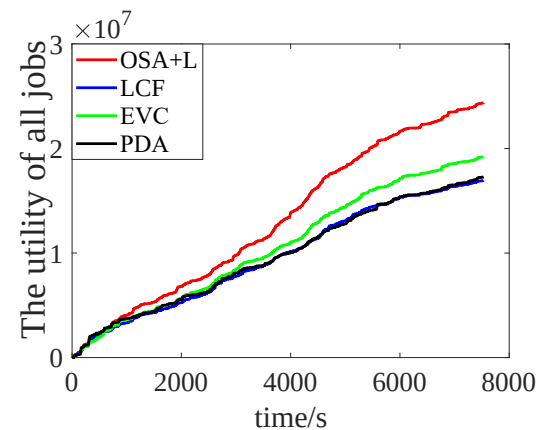


Figure 8. The total utility by OSA + L.

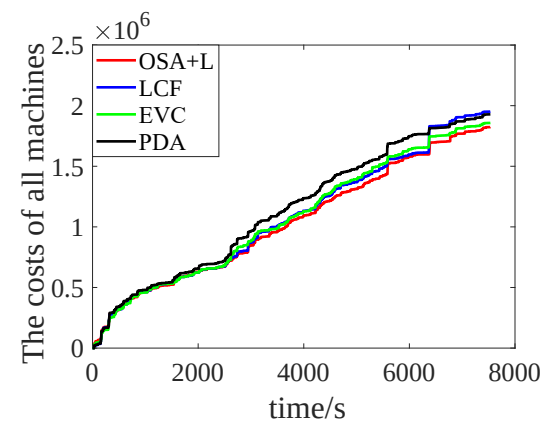


Figure 9. The total costs by OSA + L.

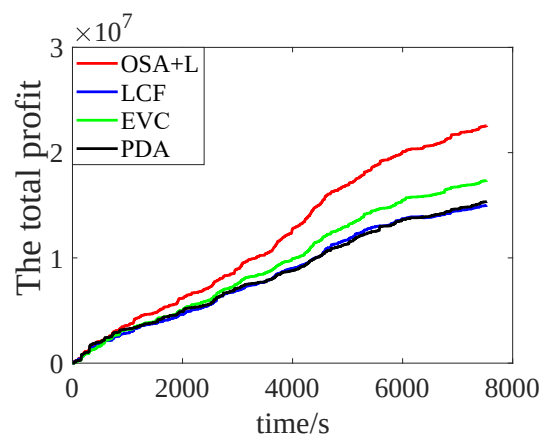


Figure 10. The total profit by OSA + L.

7. Conclusions

This work constructs a dedicated model for online optimization that simultaneously captures profit maximization and heterogeneous cluster characteristics. This work leverages a pseudo-tree architecture to implement job allocation logic. Our online scheduling algorithm updates decisions based on historical assignment results, serving as a core component to generate valid scheduling outputs. Moreover, this paper derives convex optimization analysis tools via a novel dual-fitting approach, supporting fine-grained per-step analysis through the measurement of objective value increments. We acknowledge key limitations and full known job parameters upon arrival, without stochastic uncertain workloads. Resources are treated as independent linear dimensions ignoring hardware layout coupling, with pseudo-tree topology unable to handle node failures. Future work will incorporate reconfiguration costs, stochastic robust optimization and tree restructuring to resolve these restrictions.

To verify the robustness and general applicability of our proposed framework, we use multi-angle experiments and two models to compare with a wide range of scheduling algorithms in our evaluation section. This work also adopts the preprocessed Google Cluster Trace to evaluate our scheduler on real anonymized production cloud workloads. Moreover, existing recent schedulers mostly assume lossless full-precision inter-node communication and lack hierarchical aggregation for distributed clusters. However, our pseudo-tree structure effectively reduces signaling overhead for large-scale cloud deployments. This paper carries out broad comparisons spanning mainstream scheduling paradigms and recent state-of-the-art approaches published from 2022 to 2026, including greedy heuristics, auction-based IaaS cloud resource allocation, iterative ADMM primal–dual schemes, Kubernetes DRA heterogeneous accelerator schedulers, and competitive-analysis-driven online welfare maximization algorithms. Unlike greedy baselines with $\mathcal{O}(N \cdot N)$ sorting overhead and conventional primal–dual SOTAs with multiplicative $\mathcal{O}(I \cdot N)$ complexity, our framework delivers iteration-free linear $\mathcal{O}(N)$ computational complexity.

Author Contributions: Conceptualization, Q.L.; Methodology, Q.L.; Software, X.W.; Validation, S.W.; Formal Analysis, Q.L.; Investigation, Q.L.; Writing—Original Draft Preparation, Q.L.; Writing—Review and Editing, Q.L.; Supervision, L.M. and L.C.; Funding Acquisition, W.D. All authors have read and agreed to the published version of the manuscript.

Funding: Guangdong Police College “The Eagle Project” Talent Project (2026FY02) and Guangdong Police College “Jin Eagle Scholar” Talent Project (2026JY01).

Data Availability Statement: The original data presented in the study are openly available at Google cluster-usage traces <http://www.mturk.com>.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A

Appendix A.1. Proof of Lemma 1

Proof. To prove Lemma 1, we rewrite the utility of job j as follows:

$$f_j\left(\sum_{t=a_j}^{d_j} y_j(t)\right) = f_j(y_j(t_1) + y_j(t_2) + \cdots + y_j(t_n)). \quad (\text{A1})$$

where $t_1, t_2, \dots, t_n$ denote the time slot when a task from job j is scheduled.

By the law of Lagrangian mean value, it can be seen that

$$\begin{aligned} & f_j(y_j(t_1) + y_j(t_2) + \cdots + y_j(t_n)) \\ &= f_j(y_j(t_1) + \cdots + y_j(t_{n-1})) + f'_j(\epsilon_j(t_n)) \cdot y_j(t_n). \end{aligned} \quad (\text{A2})$$

where $\epsilon_j(t_n) \in (y_j(t_1) + \cdots + y_j(t_{n-1}), y_j(t_1) + y_j(t_2) + \cdots + y_j(t_n))$

Repeating the above process, it can be seen that

$$\begin{aligned} & f_j(y_j(t_1) + y_j(t_2) + \cdots + y_j(t_{n-1})) \\ &= f_j(y_j(t_1) + \cdots + y_j(t_{n-2})) + f'_j(\epsilon_j(t_{n-1})) \cdot y_j(t_{n-1}). \end{aligned} \quad (\text{A3})$$

where $\epsilon_j(t_{n-1}) \in (y_j(t_1) + \cdots + y_j(t_{n-2}), y_j(t_1) + y_j(t_2) + \cdots + y_j(t_{n-1}))$

Thus, we get

$$f_j(y_j(t_1) + y_j(t_2) + \cdots + y_j(t_n)) = f_j(y_j(t_1)) + \sum_{t=2}^n f'_j(\epsilon_j(t_n)) \cdot y_j(t_n). \quad (\text{A4})$$

In our model, let $f_j(y_j(t_1)) = T_{jt}(y_j(t_1))$ and $f'_j(\epsilon_j(t_n)) \cdot y_j(t_n) = T_{jt}(y_j(t_n))$. Thus, $f_j(\sum_{t=a_j}^{d_j} y_j(t))$ can be formulated as follows:

$$f_j\left(\sum_{t=a_j}^{d_j} y_j(t)\right) = \sum_{t=a_j}^{d_j} T_{jt}(y_j(t)). \quad (\text{A5})$$

□

Appendix A.2. Proof of Lemma 3

Proof. To prove Lemma 3, based on Equation (6), B_1 is given by

$$\begin{aligned} B_1 = \max_{\alpha \geq 0, \beta \geq 0} \{ & \sum_{j=1}^N f_j\left(\sum_{t=a_j}^{d_j} y_j(t)\right) - \sum_{i=1}^M \sum_{t=a_j}^{d_j} g_{it}(z_i(t)) - \\ & \left(\sum_{j=1}^N \sum_{t=a_j}^{d_j} \alpha_j(t) \cdot y_j(t) - \sum_{i=1}^M \sum_{t=a_j}^{d_j} \beta_i(t) \cdot z_i(t)\right) \} \end{aligned} \quad (\text{A6})$$

It is inferred from Lemma 1 that $f_j(\sum_{t=a_j}^{d_j} y_j(t)) = \sum_{t=a_j}^{d_j} T_{jt}(y_j(t))$. Thus, B_1 can be given as follows:

$$\begin{aligned} B_1 = \max_{\alpha \geq 0, \beta \geq 0} \{ & \sum_{j=1}^N \sum_{t=a_j}^{d_j} T_{jt}(y_j(t)) - \sum_{i=1}^M \sum_{t=a_j}^{d_j} g_{it}(z_i(t)) - \\ & \left(\sum_{j=1}^N \sum_{t=a_j}^{d_j} \alpha_j(t) \cdot y_j(t) - \sum_{i=1}^M \sum_{t=a_j}^{d_j} \beta_i(t) \cdot z_i(t)\right) \} \end{aligned} \quad (\text{A7})$$

For Equations (1) and (3), it is implied that $T_{jt}(y_j(t)) = f'_j(\epsilon_j(t)) \cdot y_j(t) = h_j s(\epsilon_j(t))^{s-1} \cdot y_j(t)$ and $g_{it}(z_i(t)) = k_i(z_i(t))^m$. Therefore, the value of $B_1(\alpha_j(t), \beta_i(t))$ for job j on machine i in time slot t can be characterized by

$$\begin{aligned} B_1(\alpha_j(t), \beta_i(t)) &= T_{jt}(y_j(t)) - g_{it}(z_i(t)) - (\alpha_j(t) \cdot y_j(t) - \beta_i(t) \cdot z_i(t)) \\ &= h_j s(\epsilon_j(t_n))^{s-1} y_j(t) - \alpha_j(t) y_j(t) + \beta_i(t) z_i(t) - k_i(z_i(t))^m. \end{aligned} \quad (\text{A8})$$

Based on Lemma 3 of [8], we calculate partial derivatives $\frac{d\Phi(B_1, \alpha, \beta)}{dy_j^*(t)}$ and $\frac{d\Phi(B_1, \alpha, \beta)}{dz_i^*(t)}$ for the maximum value:

$$\frac{d\Phi(B_1, \alpha, \beta)}{dy_j^*(t)} = h_j s(\epsilon_j(t))^{s-1} - \alpha_j(t) = 0 \quad (A9)$$

$$\frac{d\Phi(B_1, \alpha, \beta)}{dz_i^*(t)} = \beta_i(t) - k_i m (z_i^*(t))^{m-1} = 0 \quad (A10)$$

Thus, it can be seen that

$$h_j s(\epsilon_j(t))^{s-1} = \alpha_j(t) \quad (A11)$$

$$z_i^*(t) = \left(\frac{\beta_i(t)}{k_i m}\right)^{\frac{1}{m-1}} \quad (A12)$$

Substituting Equations (A11) and (A12) into (A8), the maximum value $B_1^*(\beta_i(t))$ is given by

$$B_1^*(\beta_i(t)) = \left(1 - \frac{1}{m}\right) \left(\frac{1}{k_i m}\right)^{\frac{1}{m-1}} \beta_i(t)^{\frac{m}{m-1}}. \quad (A13)$$

□

Appendix A.3. Proof of Lemma 4

Proof. To prove Lemma 4, inferred from Equations (3) and (8), it can be seen that

$$g'_{it}(z_i(t)) = \frac{dg_{it}(z_i(t))}{dz_i(t)} = k_i m (z_i(t))^{m-1} \quad (A14)$$

$$\beta_i(t) = g'_{it}(\theta_i \cdot z_i(t)) = k_i m (\theta_i \cdot z_i(t))^{m-1} \quad (A15)$$

Therefore, it follows that

$$\beta_i(t) dz_i(t) - g'_{it}(z_i(t)) dz_i(t) = k_i m (z_i(t))^{m-1} (\theta_i^{m-1} - 1) dz_i(t) \quad (A16)$$

For Equation (A13), it can be seen that

$$B_1'^*(\beta_i(t)) d\beta_i(t) = \left(\frac{\beta_i(t)}{k_i m}\right)^{\frac{1}{m-1}} d\beta_i(t). \quad (A17)$$

Substituting Equation (A15) into Equation (A17), it follows that

$$B_1'^*(\beta_i(t)) d\beta_i(t) = k_i m (z_i(t))^{m-1} \theta_i^m (m-1) dz_i(t) \quad (A18)$$

Combining Equations (A16) and (A18) with Definition 1, it follows that

$$\begin{aligned} & k_i m (z_i(t))^{m-1} (\theta_i^{m-1} - 1) dz_i(t) \\ & \geq \frac{1}{\delta_j^i} (k_i m (z_i(t))^{m-1} \theta_i^m (m-1) dz_i(t)). \end{aligned} \quad (A19)$$

Calculating Equation (A19), if $m \geq 2$ then $\frac{\theta_i^m(m-1)}{\theta_i^{m-1}-1} \leq 4(m-1) < 4m$ and if $2 > m > 1$ then $\frac{\theta_i^m(m-1)}{\theta_i^{m-1}-1} \leq m^{\frac{m}{m-1}} < 4m$. Thus, we obtain

$$\delta_j^i = 4m. \quad (A20)$$

□

References

- Li, Q.; Wen, S.; Wang, X.; Du, W.; Cai, L.; Xu, H. Online Job Scheduling for Profit Maximization in a Heterogeneous Cluster. In *23rd IEEE International Symposium on Parallel and Distributed Processing with Applications*; IEEE: New York, NY, USA, 2025.
- Available online: <http://www.mturk.com> (accessed on).
- Liu, Q.; Li, Z.; Fang, Z.X. Smoothed Online Decision Making in Communication: Algorithms and Applications. *IEEE/ACM Trans. Netw.* **2024**, *33*, 463–478.
- Xu, H.; Liu, Y.; Lau, W.C.; Zeng, T.; Guo, J.; Liu, A.X. Online Resource Allocation with Machine Variability: A Bandit Perspective. *IEEE/ACM Trans. Netw.* **2020**, *28*, 2243–2256.
- Tian, Z.; Zheng, L. Scheduling Unrelated Parallel Batch Processing Machines Under Time-of-Use Electricity Prices. *IEEE Trans. Autom. Sci. Eng.* **2024**, *22*, 1631–1646.
- Wang, Y.; et al. Dynamic Resource Allocation for Heterogeneous Accelerators via Kubernetes DRA. *IEEE Internet Things J.* **2025**, 22910–22919.
- Wu, J.; et al. Distributed Resource Allocation for End-Edge-Cloud Heterogeneous Clusters via ADMM. *IEEE Trans. Mob. Comput.* **2025**.
- Zhang, X.; Huang, Z.; Wu, C.; Li, Z.; Lau, F.C. Online Auctions in IaaS Clouds: Welfare and Profit Maximization with Server Costs. *IEEE/ACM Trans. Netw.* **2017**, *25*, 10340–1047.
- Wang, Z.; Goudarzi, M.; Buyya, R. TF-DDRL: A Transformer-enhanced Distributed DRL Technique for Scheduling IoT Applications in Edge and Cloud Computing Environments. *IEEE Trans. Serv. Comput.* **2025**, *18*, 1039–1053.
- Wang, C.; Luo, Y.; Du, W.; Wang, K.; Gu, N.; Yu, J. Faster and Stronger: Unleashing Data Processing Potential through Hardware Heterogeneity. *IEEE Internet Things J.* **2025**, *12*, 14559–14576.
- Wedelin, D. An algorithm for large scale 0-1 integer programming with application to airline crew scheduling. *Ann. Oper. Res.* **1995**, *57*, 283–301.
- Elnikety, S.; Larus, J.; Yan, C.; He, Y.; Larus, J. Scheduling interactive services with partial execution. In *Proceedings of SOCC*; 2012.
- Zhang, Y.; Liang, W.; Xu, Z.; Jia, X.; Yang, Y. Profit Maximization of Delay-Sensitive, Differential Accuracy Inference Services in Mobile Edge Computing. *IEEE Trans. Mob. Comput.* **2025**, *24*, 6209–6224.
- Zhao, H.; Wang, Z.; Cheng, G.; Qian, W.; Chen, P.; Yin, J.; Dustdar, S.; Deng, S. Online Workload Scheduling for Social Welfare Maximization in the Computing Continuum. *IEEE Trans. Serv. Comput.* **2025**, *18*, 2267–2280.
- Zhang, Y.; Luo, L.; Sun, G.; Yu, H.; Li, B. Deadline-Aware Online Job Scheduling for Distributed Training in Heterogeneous Clusters. In *IEEE Trans. Cloud Comput.* **2025**, *13*, 590–604.
- Xu, L.; Gu, X.; Song, Z. Optimal charging for large-scale heterogeneous electric vehicles: A novel paradigm based on learning and backward clustering. *Appl. Energy* **2025**, *382*, 125232.
- Bao, Y.; Peng, Y.; Wu, C.; Li, Z. Online job scheduling in distributed machine learning clusters. In *IEEE INFOCOM*; IEEE: New York, NY, USA, 2018.
- You, W.; Jiao, L.; Li, J.; Zhou, R. Scheduling DDoS Cloud Scrubbing in ISP Networks via Randomized Online Auctions. In *Proceedings of IEEE INFOCOM*; IEEE: New York, NY, USA, 2020.
- Zhou, Z.; Yang, S.; Pu, L.; Yu, S. CEFL: Online Admission Control, Data Scheduling, and Accuracy Tuning for Cost-Efficient Federated Learning Across Edge Nodes. *IEEE Internet Things J.* **2020**, *7*, 9341–9356.
- Shrouf, F.; Ordieres-Meré, J.; García-Sánchez, A.; Ortega-Mier, M. Optimizing the production scheduling of a single machine to minimize total energy consumption costs. *J. Clean. Prod.* **2014**, *67*, 197–207.
- Gan, K.; Keyvanshokoh, E.; Liu, X.; Murphy, S. Contextual Bandits with Budgeted Information Reveal. In *Proceedings of AISTATS*; PMLR: Cambridge, MA, USA, 2024.
- Gupta, A.; Krishnaswamy, R.; Pruhs, K. Online primal-dual for nonlinear optimization with applications to speed scaling. In *Proceedings of WAOASpringer*; Berlin/Heidelberg, Germany, 2002; pp. 173–186.
- Devanur, N.R.; Huang, Z. Primal dual gives optimal energy efficient online algorithms. In *SODA*; 2014.
- Du, A.; Jia, J.; Chen, J.; Wang, X.; Huang, M. Online Queue-Aware Service Migration and Resource Allocation in Mobile Edge Computing. *IEEE Trans. Veh. Technol.* **2025**, *74*, 8063–8078.
- Xu, J.; Chen, G.Y.; Zhou, Y.; Rao, Z.; Yang, D.; Xie, C. Incentive Mechanisms for Large-Scale Crowdsourcing Task Diffusion Based on Social Influence. *IEEE Trans. Veh. Technol.* **2021**, *70*, 3731–3745.
- Doostmohammadian, M.; Aghasi, A.; Rikos, A.I.; Grammenos, A.; Kalyvianaki, E.; Hadjicostis, C.N.; Johansson, K.H.; Charalambous, T. Distributed CPU scheduling subject to nonlinear constraints. In *IEEE Conference on Control Technology and Applications*; IEEE: New York, NY, USA, 2022; pp. 746–751.
- Doostmohammadian, M.; Gabidullina, Z.R.; Rabiee, H.R. Machine learning and CPU (central processing unit) scheduling co-optimization over a network of computing centres. *Eng. Appl. Artif. Intell.* **2026**, *163*, 113018.
- Zheng, Z.; Shroff, N. Online multi-resource allocation for deadline sensitive jobs with partial values in the cloud. In *Proceedings of IEEE INFOCOM*; IEEE: New York, NY, USA, 2016.

29. Zheng, Z.Z.; Shroff, N.B. Online Welfare Maximization for Electric Vehicle Charging with Electricity Cost. In *ACM e-Energy*; Association for Computing Machinery: New York, NY, USA, 2014.
30. Zhao, Y.; Tian, C.; Zhu, Z.; Cheng, J.; Qiao, C.; Liu, A.X. Minimize the Make-span of Batched Requests for FPGA Pooling in Cloud Computing. *IEEE Trans. Parallel Distrib. Syst.* **2018**, *29*, 2514–2527.
31. Xu, Z.; Qiao, H.; Liang, W.; Xu, Z.; Xia, Q.; Zhou, P.; Rana, O.F.; Xu, W. Flow-Time Minimization for Timely Data Stream Processing in UAV-Aided Mobile Edge Computing. *ACM Trans. Sens. Netw.* **2024**, *20*, 1–28.
32. Ma, G.; Li, H.; Wang, X.; Chen, X.; Bian, Y.; Hu, M.; Wang, X.; Zhang, J. Mobility-aware Task Splitting and Resource Allocation for Vehicular MEC. In *ACM MobiCom*; Association for Computing Machinery: New York, NY, USA, 2022.
33. Mount, D.M. *CMSC 451: Greedy Algorithms for Scheduling*; University of Maryland Technical Report; 2025.
34. Boyd, S.; Vandenberghe, L. *Convex Optimization*; Cambridge University Press: Cambridge, UK, 2004.
35. Reiss, C.; Wilkes, J.; Hellerstein, J.L. Google Cluster-Usage Traces. May 2011. Available online: <https://github.com/google/cluster-data> (accessed on 1 June 2026).

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.