Joint Offloading and Service Selection via Matching and Auction Theory for Multi-Task Dependent Computation-Intensive Applications

Benedetta Picano[a], Marco Paolieri[b], Laura Carnevali[a], Enrico Vicario[a]

[a] *University of Florence, Department of Information Engineering, Via di Santa Marta 3, Florence, 50139, Italy*
[b] *University of Southern California, Department of Computer Science, 941 Bloom Walk, Los Angeles, 90089, CA, USA*

Abstract

The increasing complexity of next-generation services demands efficient orchestration across the edge-to-cloud continuum to balance computational intensity, latency constraints, and resource availability. These services are typically decomposed into interdependent sub-tasks, requiring careful synchronization to meet stringent completion time requirements. The challenge is further amplified in heterogeneous and resource-constrained edge environments, where multiple providers dynamically compete for sub-task execution. This paper introduces a game-theoretic stochastic framework that optimizes system welfare from both users' and providers' perspectives, ensuring efficient task allocation across distributed computing resources. We propose a Cumulative Distribution Function (CDF)-driven game, where edge nodes serve as intermediaries between users and cloud/edge service providers. The framework is structured as a two-level mechanism: (i) a matching game governing the user-to-edge node association, and (ii) a nested Vickrey-Clarke-Groves auction selecting the optimal provider, based on a CDF-driven assessment of service completion times. To enhance feasibility in decentralized edge computing environments, provider bids are represented as uniform CDFs, establishing a dominance relation that mitigates strategic manipulation. We theoretically analyze cheating strategies, showing that truthful bidding is a rational provider behavior and that the resulting user–edge matching satisfies a suitable stability notion. Extensive simulations compare the proposed approach against a full-knowledge-based allocation, conventional game-theoretic models, and a heuristic recently proposed in the literature, evaluating the price of anarchy, system welfare, and outage probability. The results demonstrate the effectiveness of our framework in achieving resilient, cost-efficient, and low-latency orchestration across the edge-to-cloud continuum in heterogeneous edge deployments.

Keywords: service composition, service selection, auction theory, matching theory

The edge-to-cloud continuum has emerged as a fundamental enabler of next-generation intelligent systems. This paradigm integrates edge computing (EC) and cloud infrastructures to provide seamless, adaptive, and efficient processing capabilities for increasingly complex applications, including artificial intelligence (AI)-driven services. In this ecosystem, edge nodes act as intermediaries, handling latency-sensitive and computation-intensive tasks before offloading residual workloads to the cloud for large-scale optimization [1, 2, 3, 4, 5].

Among the most demanding AI-based applications, multi-modal learning, large-scale language models, virtual and augmented reality, and autonomous systems require extensive data processing before meaningful insights can be extracted from their embedding space [6, 7]. Unlike traditional networking scenarios where communication delays often dominate, the processing time for generating actionable intelligence from high-dimensional data embeddings has become the primary bottleneck.

These AI-driven services typically consist of complex workflows, where each sub-task has strict interdependencies that dictate concurrency and synchronization constraints [6, 8]. A canonical example is real-time face recognition, which comprises multiple processing steps: object acquisition, face detection, feature extraction, and classification [9, 8, 10]. In contemporary distributed architectures, each sub-task of the workflow is typically implemented as an independent microservice under the Service-Oriented Architecture (SOA) paradigm [11, 12]. We refer to the entity implementing a sub-task as a provider, i.e., a logical service com-

ponent exposing the functionality of that sub-task. Edge nodes orchestrate workflow execution by selecting, for each sub-task, the most suitable provider among those available. Service execution is then governed by Service Level Agreements (SLAs) established between customers and providers [13, 14]. SLAs define the expected Quality of Service (QoS) through Service Level Objectives (SLOs), while the actual performance delivered is captured by Service Level Indicators (SLIs). These agreements define the QoS requirements, which typically include hard or soft deadlines on service completion times. While an SLO is a predefined benchmark that establishes expected service quality, the SLI quantifies the actual uptime and operational effectiveness of the system, serving as a compliance measure against the agreed SLA. In dynamic edge-to-cloud environments, where service execution is influenced by network heterogeneity, fluctuating workloads, and resource availability, ensuring SLO adherence requires intelligent orchestration and adaptive resource allocation strategies.

Despite the increasing reliance on distributed computing infrastructures, most existing approaches oversimplify service orchestration, either neglecting sub-task dependencies or assuming idealized execution environments. However, the failure to explicitly model and optimize resource-constrained task execution leads to suboptimal performance, SLA violations, and significant operational inefficiencies [9, 8]. Optimized service placement and workload distribution are therefore crucial to mitigating these inefficiencies while ensuring scalability, robustness, and fairness in resource allocation.

This paper tackles the joint problem of offloading and service selection in edge-to-cloud environments, addressing the interplay between users, edge nodes, and computational providers. In our framework, users announce their workflows and specify SLOs through the Cumulative Distribution Function (CDF) of the end-to-end (e2e) service completion time. Edge nodes mediate the assignment of sub-tasks to providers, which, in turn, submit completion time bids expressed as CDFs. To enable scalable, low-complexity orchestration, we adopt matching theory and auction mechanisms, ensuring optimal yet lightweight decision-making for dynamic service provisioning.

Our proposed stochastic framework leverages: *i)* matching game to model user-edge node allocation, capturing the interdependencies within the allocation process; *ii)* a nested Vickrey-Clarke-Groves (VCG) auction for provider selection, enforcing truthful bidding strategies and discouraging manipulative behaviors. By casting providers' CDF bids into uniform distributions, we reduce computational overhead while enabling safe predictability of sustainable SLOs. This transformation allows us to model the problem as a game with partial information, ensuring adaptability to dynamic system conditions [15].

Our main contributions are summarized as follows.

- Joint formulation of offloading and service selection within an edge environment, explicitly considering sub-task dependencies. We prove the NP-hardness of the problem and propose a game-theoretic framework based on matching theory and auctions to ensure efficient resource utilization.
- Development of a novel low-complexity CDF-driven VCG auction for provider selection, where bids are expressed as completion time distributions rather than fixed values. This enables predictable SLO enforcement and mitigates provider cheating strategies.
- Comprehensive performance evaluation, comparing the proposed framework against full-knowledge approaches and other baselines, including one decoupled optimization strategy, analyzing metrics such as price of anarchy, system welfare, and outage probability.

This work provides a scalable, distributed, and adaptive approach to service orchestration in AI-driven edge-to-cloud environments, ensuring low-latency, high-efficiency execution of computation-intensive workflows.

In the rest of the paper, the problem statement is detailed and the solution approach is outlined in Section 1, and a review of related literature is presented in Section 2. The auction scheme and the matching game are presented in Section 3 and Section 4, respectively. Performance analysis is reported in Section 5, and conclusions are finally drawn in Section 6.

1. Problem Statement and Approach

1.1. System Model

We consider a system model with three types of participants:

- a set $\mathcal{C} = \{1, ..., C\}$ of users, each requiring repeated execution of a workflow to be completed with a required distribution F_c of the e2e completion time;
- a set $\mathcal{P} = \{1, ..., P\}$ of antagonistic providers, each representing, in the SOA vision, an *atomic service entity* that offers the implementation of a specific sub-task composing the workflow. Each provider p is characterized by an agreed and an actual distribution of the completion time, as observed through repeated executions [12, 11].
- a set of resource-constrained edge nodes $\mathcal{A} = \{1, ..., A\}$ which intermediate the relation between users and providers; to this end, each edge node a has access to a subset of available providers $\mathcal{P}_a \subseteq \mathcal{P}$, and it assumes the responsibility to select a provider for each sub-task in each workflow, to manage SLAs on both sides towards users and providers, and to orchestrate service delivery. In this sense, edge nodes act as computational orchestrators with logical access to service interfaces, rather than physical co-location with provider infrastructures.

The heterogeneous computing environment considered in this work consists of a hierarchical edge infrastructure with nodes of varying computational capabilities. Lightweight edge devices, such as gateways or embedded systems, primarily handle data acquisition, filtering, and pre-processing tasks under strict resource limitations. In contrast, micro-edge servers operate as small-scale edge data centers located in close proximity to end users, equipped with multi-core CPUs, moderate GPU acceleration, and enhanced memory and storage capacity. Their virtualization layer supports the deployment of containerized applications or microservices, which can be dynamically orchestrated according to workload variations and latency constraints. Each edge node a can be selected by multiple users under the limit of a maximum resource capacity L_a that constrains the total complexity of served workflows; each sub-task can be delivered by multiple providers with equivalent functionality but different qualities.

In this service-oriented model, each sub-task of the workflow is implemented by an independent microservice-like entity, i.e., a provider. Providers expose the logic of a specific sub-task as a logical service interface, while edge nodes act as orchestrators that dynamically compose these services to execute the end-to-end workflow requested by the user. Without loss of generality, the model assumes that each user requires a single workflow and each provider delivers a single sub-task. Multiplicity can be accommodated in the framework by introducing multiple virtual users and providers, as also in [16]. Finally, for simplicity but open to relaxation, each workflow is requested by a single user. Note that a multiplicity of workflows must be repeatedly executed for a number of times sufficient to let stable statistics of service completion time emerge.

On the one hand, the SLO in the agreement among an edge node and the provider of each sub-task s is expressed as a CDF distribution $F_Y(\cdot)$ identified through the VCG-inspired second-price rule presented in Section 3, and the SLI of each sub-task is the statistics of actual completion times t_D, emerging in repeated executions. According to this, we assume that the edge node a will pay the provider selected for sub-task $s \in S$ a reward $\mathcal{U}_{a,s}$. The term $F_Y(t_D)$ is the cumulative distribution function of the completion-time random variable Y, evaluated at the observed completion time t_D. The parameter K is a constant baseline term introduced to regulate the overall reward level. For simplicity, we assume that all sub-tasks share the same baseline parameter K. The specific choice of K will be discussed later when analyzing the incentive properties of the mechanism in Section 3.1. The treatment can be easily extended to encompass heterogeneous baseline parameters depending on the service type or its position within the workflow topology. Let $f(\cdot)$ be an increasing function mapping QoS performance into a utility contribution. For analytical tractability, we adopt the linear specification $f(x) = x$. Thus, we define the provider reward as

$$\mathcal{U}_{a,s} = f(1 - F_Y(t_D)) + K. \tag{1}$$

On the other hand, each user expresses its required SLO as a CDF F_c on the e2e workflow completion time t_E, and the SLI supplied by the selected edge node is the statistics of actual e2e completion times measured in repeated executions of the workflow. For the edge node, the workflow execution subtends an outlay cost for the orchestration of services and the reward of providers. We term this quantity R_c. To cover the cost R_c, the user pays the edge node a reward b_c, with $b_c > R_c$, which accounts for the functional complexity of the supplied workflow, multiplied by a factor that accounts for the delivered QoS, here expressed in terms of the e2e time provisioned with respect to the SLO posed by user c. According to this, the edge node utility is expressed as

$$\mathcal{U}_{c,a} = (1 - F_c(t_E) + \Theta) \cdot (b_c - R_c), \tag{2}$$

where t_E is the actual e2e workflow time, and $\Theta > 0$ is an additive corrective factor to induce positive rewards. The multiplicative structure in (2) reflects the joint dependence of the utility on both profit and delivered QoS. The term $(1 - F_c(t_E) + \Theta)$ acts as a QoS-related weight, modulating the economic margin $(b_c - R_c)$ according to the achieved service performance. High utility is obtained only when the workflow yields a positive margin $(b_c - R_c)$ and satisfies the latency requirements, whereas poor performance in either dimension proportionally reduces the overall utility. This formulation captures the complementarity between efficiency and profitability and enforces a balanced incentive mechanism, preventing situations in which one component alone dominates the utility. The computationally intensive nature of next-generation services, particularly those based on AI and Large-Scale Data Processing, poses a significant challenge for the edge-to-cloud continuum. If not properly optimized, computation can become the primary bottleneck in modern applications. This shift in computational demand is further accentuated by the emergence of high-velocity network infrastructures. Note that, in scenarios where the communication delay is non-negligible, the e2e delay can be modeled as a two-node cascade system, where the first subsystem represents the transmission channel and the second subsystem accounts for the computational processing. However, in the considered edge-to-cloud continuum, we assume a well-engineered network infrastructure designed to guarantee negligible communication latency between interacting entities.

This paper builds upon these insights, developing an optimized edge resource allocation strategy that focuses on processing time, assuming that transmission delays are negligible. This approach allows for an effective orchestration of AI-driven services, minimizing processing bottlenecks while ensuring seamless system performance.

1.2. Problem Formulation

The objective of this paper is the maximization of a measure of system welfare accounting for joint rewards of providers and edge nodes in the answer to the demand specified by users. The inherent interdependence between user offloading and service selection requires a joint optimization approach, since the efficiency of the offloading decision directly depends on the QoS obtained, via providers, at the selected edge node, and conversely, the user–edge association pattern shapes the subsequent service selection dynamics. However, a decoupled implementation is also considered in the performance analysis to quantify the impact of neglecting such coupling and to provide a meaningful comparison baseline. The optimization problem can be expressed as:

$$\max_{\mathbf{\Gamma},\mathbf{\Delta}} \sum_{c=1}^{C}\sum_{a=1}^{A} \mathcal{U}_{c,a}\gamma_{c,a} + \sum_{s=1}^{S}\sum_{p=1}^{P} \mathcal{R}_{s,p}\delta_{s,p}, \tag{3}$$

$$s.t. \quad \sum_{p=1}^{P} \delta_{s,p} \leq 1, \quad \forall s \in \mathcal{S} \tag{4}$$

$$\sum_{a=1}^{A} \gamma_{c,a} \leq 1, \quad \forall c \in \mathcal{C} \tag{5}$$

$$\sum_{c=1}^{C} R_c\gamma_{c,a} \leq L_a, \quad \forall a \in \mathcal{A} \tag{6}$$

where $\mathbf{\Gamma}$ is the assignment matrix with $\gamma_{c,a}$ equal to 1 when edge node a is selected to serve user c, and 0 otherwise. And, similarly, $\mathbf{\Delta}$ is the selection matrix with $\delta_{s,p}$ equal to 1 when sub-task s is outsourced to provider p and 0 otherwise. $\mathcal{R}_{s,p}$ is the reward of provider p in completing sub-task s, as described in Section 3. In the formulation, when the same sub-task belongs to more than one workflow, sub-task occurrences are handled as different sub-task, and a number of rows equal to the number of service occurrences is added to $\mathbf{\Delta}$.

Constraint (4) imposes that each sub-task s can be outsourced to one and only one provider, and constraint (5) points out that each user is assigned to at most one edge node. Eq. (6) requires that resources allocated to the edge node a cannot exceed its maximum resource capacity L_a.

The formulated problem is NP-hard, as it can be reduced to the 0-1 knapsack problem [17]. Specifically, by considering a simplified case where the reward maximization for providers is disregarded and by setting $A = 1$, the optimization problem reduces to maximizing the objective function $\max \sum_{c=1}^{C} \mathcal{U}^1 \gamma_{c,1}$, subject to the constraint $\sum_{c=1}^{C} R_c \gamma_{c,1} \leq L_1$. Given that $\gamma_{c,1}$ is binary, it is possible to map L_1 in the knapsack capacity, and $\mathcal{U}^1$ and R_c as the weight and volume of the generic item, respectively, confirming the inherent computational complexity of the original problem [17].

To avoid the complexity of the exact solution, this paper proposes a suboptimal scheme consisting of a matching game with externalities integrated with an auction mechanism to jointly assign users to edge nodes and outsource sub-tasks to providers. Experimental evaluation of suboptimality is developed in Section 5.

2. Related Works

Service selection for guaranteeing QoS constraints is addressed in a rich literature. Web service selection is also addressed in [18], in presence of simultaneous composite service requests. Two simultaneous auction algorithms have been applied to model the underlying integer linear programming problem, investigating both the full and the partial assignment. In paper [9], a convex programming optimization algorithm has been developed to solve the joint problem of offloading dependent tasks and service caching to minimize the makespan within a mobile edge landscape. As in paper [9], also authors in [8] develop a dependency-aware heuristic to solve both the offloading and the caching problems. Differently, deep learning has been applied in [19], where a sequence-to-sequence neural network and the proximal policy optimization technique are exploited to provide a data-driven solution to the offloading problem in presence of dependent tasks within a cooperative vehicular network. Differently, a dependency-based clustering algorithm has been developed in [20] to enhance the flexibility of power systems. In paper [21] the task offloading problem in vehicular fog computing networks has been addressed proposing a federated learning supporting a deep Q-learning technique for optimal offloading of tasks in a collaborative computing paradigm. The approach considers both the latency and energy consumption. The problem of task offloading and scheduling in presence of dependencies has been also addressed in [22], where dynamic collaborative networks have been studied. In particular, a data-driven reinforcement learning approach has been adopted. Similarly, an actor-critic algorithm has been proposed in [23] for offloading with dependent tasks. Graph neural networks have been proposed in [24] to solve the offloading and scheduling problem. As a common trait, these works do not consider multiple resource-constrained edge nodes or service selection. Furthermore, existing works involving task dependencies typically apply data-driven approaches as machine learning, without providing a quantitative analysis of the problem addressed. Moreover, the service selection problem in the case of resource-constrained EC infrastructure with dependencies is rarely investigated. In [25], a two-stage auction mechanism is designed to perform price discovery in the first step, and optimal bidder allocation during the second phase. The main purpose is the buyer utility maximization, under assumptions of a competitive market, due to the presence of multiple providers. A multi-attribute combinatorial double auction is the focus of [26], along with the fairness and robustness. Fairness is considered through an egalitarian social welfare metric which favors the low value bids in the long term. Robustness is provided introducing provider reputation, in order to discourage cheating by the provider in form of false QoS guarantee. A multi-seller and multi-buyer double-auction scheme is also proposed in [27] to properly select cloud federations, arranged to perform service delivery. In such a scenario, the framework operates considering heterogeneous resources and

it aims at preserving features as truthfulness, individual rationality and budget balance for both the agents involved in the auction game. Likewise, a buyer cooperative strategy is developed in [28], in which an auction game is formulated to model the task offloading resource-demanded application problem into a mobile cloud computing environment. The auction mechanism is designed in a distributed fashion, and its objective is twofold: to produce a fair task allocation and to determine resource prices. Similarly, auction theory is applied in [29], where a combinatorial auction is designed within a CC environment. Authors proposed a time-varying optimization of system efficiency in provisioning heterogeneous virtual machines. The designed online auction framework results computationally efficient and truthful, guaranteeing competitive social welfare ratio for the considered scenarios.

A multiseller multibuyer double auction mechanism is developed in [30] to rule the service offloading assignment within an edge computing landscape. The scheme proposed is movement-aware since existing social interactions among mobile users are taken into account. An edge computing infrastructure is considered also in [31], where a multiparticipant double auction for the joint assignment of resources and pricing between service and providers is developed.

Uncertainty about agent quality has been studied in [32], where a procurement auction is detailed, and the agents provide estimates about quality they intend to produce. Then, agents and services are matched, and the auctioneer observes the produced QoS and performs a payment to compensate service delivery costs. Resource procurement in CC is addressed by authors in [14], in which a multi-parameters reverse auction is designed to include in the bargaining process price as well as reputation and quality measurements.

While several studies have proposed auction-based mechanisms that account for e2e service-time constraints [26], to the best of our knowledge this is the first work to introduce a CDF-driven sub-task selection scheme in an edge environment, explicitly considering soft-deadline constraints and antagonistic service providers. Beyond the sub-task selection problem, the paper develops a unified stochastic framework that jointly optimizes user–edge node association and provider selection. Specifically, we formulate a stable matching game with externalities and integrate a nested VCG auction mechanism, ensuring strategy-proof provider selection while capturing inter-task dependencies. This formulation introduces a principled coupling between offloading and service selection through CDF-based bids, a feature that is absent in prior work, explicitly modeling how the distributional guarantees offered by providers influence the overall system performance. Unlike existing approaches that treat computation and service provisioning as separate auction steps [28, 18, 27], our framework integrates these two decisions into a unified stochastic mechanism driven by distributional (CDF) information.

3. Auction-based Service Selection

3.1. A Low Complexity Vickrey-Clarke-Groves Auction

VCG auction is a Nobel-prize winning framework that provides mathematically tractable solutions to the assignment problem, aiming at simultaneously considering the perspectives of opposite parties involved in the bargaining process. VCG defines the second price auction mechanism, where the auction winner is the bidder offering the best bid, but the winning bidder is paid the amount of the second-best bidder. In our case, the bid is expressed in terms of the earliness that the provider can guarantee, and this is concretely specified as a CDF of the sub-task time $F_s(\cdot)$. Then, the auctioneer edge node collects the bids from the antagonist providers and elects the winner on the basis of the received CDFs.

The mechanism subtends the need to identify an order between the best and the second best bid, i.e., an order among CDFs, for which we consider the *Pairwise-comparison dominance* relation:

Definition 1. Let X, Y be independent random variables with probability density function (pdf) $f_X(t)$, $f_Y(t)$, and CDFs $F_X(t)$, $F_Y(t)$, respectively. X and Y are in relation of *Pairwise-comparison dominance*, denoted by $X \preceq Y$, when $Prob\{X \leq Y\} \geq \frac{1}{2}$, i.e., when

$$\int_{t=0}^{\infty} (1 - F_Y(t)) \cdot f_X(t) dt \geq \frac{1}{2}. \tag{7}$$

The scheme encounters various complexities: providers must be able to estimate the CDFs they claim, and they have to transfer their values to edge nodes. To circumvent both limitations, the provider is assumed to cast the claimed CDF into a parametric form with bounded support in a class of distributions for which the relation of pairwise dominance $\preceq$ satisfies the property of transitivity.

We assume that claimed CDFs are cast by the provider to uniform distributions. In probabilistic risk assessment literature, the uniform distribution is commonly recommended to approximate a CDF when no a priori information about its shape exists, and complexity needs to be reduced [33]. This is often grounded on the Laplace's "principle of insufficient reason" as the uniform distribution leads to the most conservative estimation about uncertainty within a bounded support [33]. Uniform distribution is also often advocated as an elective distribution when only the minimum and the maximum values are available [33] [34]. It is relevant to note that restricted to the class of uniform distributions, $\preceq$ is transitive, which is necessary for the designed scheme to guarantee the efficient and consistent selection of a best and a second best bid. In turn, transitivity ensures that $\preceq$ becomes a total order, being *transitive*, *reflexive*, and *total*. This ensures the ability to optimally select the best and the second best CDF, in compliance with Eq. (7).

In this reference, we assume that a provider, able to deliver a sub-task service with general duration X, where GEN denotes a general service-time distribution with arbitrary shape and support contained in $[a, b]$, submits as bid the uniform duration $X^{uni(a,b)}$. The extremes a and b are derived so that $X^{uni(a,b)}$ is the most competitive uniform distribution that prevents the possibility of any loss for the provider itself. This condition is satisfied by any solution of the equation

$$\int_a^b (1 - F_{X^{uni(a,b)}}(x)) \cdot f_X(x)dx = \frac{1}{2}. \tag{8}$$

Note that (8) can be developed by parts into

$$\frac{1}{(b-a)} \int_a^b F_X(x)dx = \frac{1}{2}. \tag{9}$$

Also, note that given the CDF F_X, equation (9) determines only one of the parameters a, b that identify the optimal cast of the CDF into a uniform bid. In the experimentation of Section 5, without loss of generality, we assume that the cast is performed by providers, based on a specified value of the coefficient of variation c_V.

Provider CDFs received by the edge node are ranked according to pairwise stochastic dominance. Each sub-task is then assigned to the bid that attains the minimum under this order, i.e., any bid $X_\star$ satisfying $X_\star \preceq X_i$ for all other received bids. If multiple bids are equivalent minima, the tie is broken through any deterministic rule or, alternatively, via randomized selection (e.g., based on higher-order moments), without affecting the subsequent analysis.

Summarizing, for each sub-task s of a workflow, the algorithm proceeds through the following steps.

1. The edge node performs sub-task auctioning to outsource sub-task exploitation.
2. Each provider competes to win sub-task delivery by proposing the minimum and maximum $< a, b >$ of the uniform service time distribution that it is able to offer.
3. The edge node receives bids from providers. The provider $p_\star$ with the best bid, i.e., the bid $X_\star$ such that $X_\star \preceq X_i$ for any other received bid, on the basis of Definition 1, is selected.
4. The winning bidder provides service and the auctioning computes the corresponding utilities considering the actual duration of the service in repeated executions and the second best bid, on the basis of the pairwise-comparison dominance relation according to Eq. (1).
5. The edge node computes the CDF of the actual e2e time t_E, i.e., $F_{t_E}(t_E)$, on the basis of the sub-task CDFs received by providers.

As a crucial detail on step 4), the parameter K in Eq. (1) is set equal to $-\frac{1}{2}$, so as the expected reward for the winner is:

$$E[\mathcal{R}_{s,p_\star}] = \int_{t=0}^{\infty} (1 - F_Y(t)) \cdot f_{\tilde{X}_\star}(t)dt - \frac{1}{2}, \tag{10}$$

where F_Y is the CDF of the second best bid and $f_{\tilde{X}_\star}$ is the actual pdf provided by the winning bidder, i.e., the SLI corresponding to the QoS effectively delivered.

3.2. VCG Consequences and Cheating Strategy

We evaluate the consequences of the designed mechanism when the assigned sub-task is repeated a number of times sufficiently large so as to let emerge stable statistics of the observed QoS provided by the winning bidder $p_\star$. The impact depends on the actual SLI implemented by the provider, i.e., $\tilde{X}_\star$, and the SLO of the second best bid $X_\star$. If the QoS provided by the winning bidder actually satisfies the bid, i.e., $f_{\tilde{X}_\star}(t) = f_{X_\star}(t)$, then $E[\mathcal{R}_{s,p_\star}] \in [0, \frac{1}{2}]$, with a value that is as higher as the probability that the SLI of the winner provider is lower than the SLO of the second best provider $\tilde{X}_\star$. Note that, by definition, the reward $\mathcal{R}_{s,p_\star}$ is the complement to 1 of a CDF and thus belongs to $[-\frac{1}{2}, \frac{1}{2}]$.

The break-even condition occurs if the second best bid Y is equivalent to the best bid $X_\star$, which may occur in a scenario of high competitiveness. In this case, $\int_{t=0}^{\infty}(1 - F_Y(t)) \cdot f_Y(t) \cdot dt = \frac{1}{2}$, which implies

$$E[\mathcal{R}_{s,p_\star}] = \int_{t=0}^{\infty}(1 - F_Y(t)) \cdot f_{X_\star}(t)dt - \frac{1}{2} = 0 \tag{11}$$

On the other hand, the maximum expected reward $\frac{1}{2}$ occurs when the second best bid Y is deterministically later than the SLI of the winning provider X, which occurs when $F_Y(t) > 0 \rightarrow F_{X_\star}(t) = 1$, i.e., the support of $X_\star$ is *before* [35] the support of Y.

If the actual QoS provided by the winner is better than its bid, i.e., if $\tilde{X}_\star \preceq X_\star$, then the winner improves its advantage with respect to the second best bid Y, which will result in a higher value of $E[\mathcal{R}_{s,p_\star}]$, but always under the maximum limit of $\frac{1}{2}$.

Conversely, the expected reward $E[\mathcal{R}_{s,p_\star}]$ falls under 0 if the actual QoS $f_{\tilde{X}_\star}(t)$ lowers so much it does not dominate the second best bid Y, i.e.,

$$\int_{t=0}^{\infty}(1 - F_Y(t)) \cdot f_{X_\star}(t)dt \leq \int_{t=0}^{\infty}(1 - F_Y(t)) \cdot f_Y(t) \cdot dt. \tag{12}$$

In the limit case that the actual SLI $\tilde{X}$ of the winner is deterministically later than that of the second best bid Y, i.e., if Y is *before* $\tilde{X}$ [35], then the reward of the winner becomes $-\frac{1}{2}$. In any case, the reward cannot fall under $-\frac{1}{2}$, being $\int_{t=0}^{\infty}(1 - F_Y(t)) \cdot f_{\tilde{X}_\star}(t)dt$ lower bounded by 0.

In this picture, these reward bounds also depend on the relation between the SLI that a provider can deliver and the CDF that it can claim in the bid, which is when the mechanism of the VCG auction comes into play. To this end, we define as *rational* a provider that attempts to win all and only the sub-task outsourcings for which it can have any gain without risking any loss. In light of these assumptions, the following Lemma can be stated.

Lemma 1. A rational provider will claim a truthful CDF, i.e., a CDF that it is able to satisfy.

Lemma 1 affirms that strategic provider behavior is discouraged. When a provider offers a CDF less competitive than the actual SLO that it can guarantee, it does not obtain any advantage. In fact, it will reduce the possibility to be the auction winner. Conversely, if a provider attempts to cheat by submitting an over-optimistic CDF that reflects an unrealistically favorable completion-time distribution, two cases may arise. (i) If the second-best bid is still worse than the realistic CDF, the cheating provider gains no advantage, as the auction outcome remains unchanged; (ii) If instead the second-best bid is better than the true CDF, the provider may win the auction undeservedly, but it will incur a loss because the delivered SLI will not dominate the payment CDF. Hence, the mechanism is inherently robust to cheating attempts, since no provider can anticipate the competitiveness of the second-best bid and any deviation from truthful bidding exposes the bidder to negative rewards.

The interaction mechanism among edge nodes and users is characterized by parameter Θ of Eq. (2), which rules the degree of compliance requested to the edge node, which in turn determines the degree of compliance that the edge node must request to its providers:

Behavioral Consequence (C1): *The value of Θ represents the critical value of the proposed strategy. Necessarily, $\Theta \in [\frac{1}{2}, 1]$, but an over-pessimistic choice about Θ, i.e., $\Theta = \frac{1}{2}$, may penalize the edge node reward, since it may not cover the outlay cost employed to pay providers when sub-task is completed later than the agreed SLO. Conversely, $\Theta = 1$ represents an over-optimistic case that takes away responsibility in selecting reliable providers from edge nodes.*

The value assigned to Θ also rules the level of competitiveness imposed by edge nodes to providers, controlling race conditions among providers imposed by edge nodes, and consequently the profit margin of edge nodes. In particular, when $\Theta = \frac{1}{2}$, the edge node is required to maintain a high level of competitiveness among its providers for each sub-task so that the second bid CDF is close to the best one, whereas $\Theta = 1$ implies soft competition conditions.

4. Matching Users with Edge Nodes

Matching theory offers a powerful framework to associate elements from two opposite sets, capturing how each party evaluates potential partners and enabling a balanced trade-off between their preferences. Moreover, the matching theory naturally fits a distributed system, since it involves exclusively local utility metrics to build individual preferences. In our case, the matching game is formulated among the sets of users $\mathcal{C}$ and edge nodes $\mathcal{A}$, in order to establish relations reciprocally advantageous for all the players [15], taking into account their preferences. Preference relations describe the level of interest of each element of a set in being matched with each element of the opposite set. In the following, we define the utility functions involved in the construction process of preference lists.

4.1. User Preference List

Since edge nodes have limited resources, users incur the risk of not receiving service. From the user's perspective, the individual utility is designed to avoid as much as possible such a condition, having the chance to receive the service with a quality appropriate to its SLO. Accordingly, the user c, demanding R_c resources, can receive support from edge nodes belonging to $\mathcal{A}_c$ having enough available resources $\mathcal{L}_a$. On the contrary, an edge node that does not have sufficient resources to host c cannot offer support to that user, denying service to c. We thus set the user preferences with the objective of minimizing the probability of the user being rejected, defining the individual user utility function $H_c(a)$ as

$$H_c(a) = \mathcal{L}_a - R_c, \tag{13}$$

where $\mathcal{L}_a$ is the amount of resources available on edge node a, i.e.,

$$\mathcal{L}_a = L_a - \sum_{c \in \mathcal{C}} R_c \gamma_{c,a}. \tag{14}$$

Preferences are sorted in descending order to maximize the acceptance probability for user c on edge nodes. The most preferred edge node $a^\star$ is thus

$$a_c^\star = \arg\max_{a \in \mathcal{A}_c} H_c(a). \tag{15}$$

Eq. (15) expresses that the most preferred $a_c^\star$ is the edge node having the highest number of available resources. In this reference, it is important to note that as the algorithm execution proceeds, the number of users allocated to edge nodes grows.

4.2. Edge Node Preference List

On the other hand, the preference list of each edge node a is built preferring the user $c^\star$ that maximizes the mean reward of a. In the light of received provider bids, each edge node is able to determine the proper

user. Defining the set of users proposing to the edge node a as $\mathcal{C}_a$, the corresponding edge node preference list $E_a(c)$ is built ranking in descending order the following metric

$$E_a(c) = (b_c - R_c)\left[\int_{x=0}^{\infty} f_{\tilde{X}_\star}(x)(1 - F_c(x))dx - \frac{1}{2}\right], \tag{16}$$

which represents the expected edge node utility, where the term Θ is neglected to be a constant value, equal for all users. Thus the most preferred $c_a^\star$ is given by

$$c_a^\star = \arg\max_{c \in \mathcal{C}_a} E_a(c). \tag{17}$$

Note that this subtends the ability of a to efficiently aggregate CDFs of individual sub-task to derive the e2e CDF. In our implementation, this is achieved by exploiting the ORIS tool [36, 37]. Besides, the assignment of users to edge nodes is solved here through a modified version of the Gale-Shapley algorithm (GSA) [15]. Summarizing, the assignment mechanism among users and edge nodes acts as follows.

1. Each unassigned user $c \in \mathcal{C}$ builds its preferences list on edge nodes according to Eq. (13);
2. Each edge node a, receiving a set of proposals $\mathcal{C}_a \subseteq \mathcal{C}$, builds its preference list performing, for each user, the VCG auction;
3. Each edge node that receives more than one proposal, i.e., $|\mathcal{C}_a| > 0$, selects the most favorite user $c_a^\star$ in accordance with Eq. (17), and accepts to serve $c_a^\star$ among those received, rejecting the others;
4. Each unassigned user $c \in \mathcal{C}$ deletes from the possible edge nodes those having an available resource capacity lower than R_c;
5. Steps 1 to 4 are repeated until all the users have been matched with one edge node.

The developed matching algorithm terminates in a finite number of iterations. It has also the following implications, in terms of user fully-rational perspective, and provider alternation in winning auction:

Behavioral Consequence (C2): *Selfish user behaviors are allowed. This is the practical consequence of not having bounded above the term b_c. As a matter of principle, each user may select an arbitrary large $b_c > R_c$ in order to be assigned to its most preferred edge node.*

Behavioral Consequence (C3): *The proposed market does not safeguard alternation in sub-task adjudication among providers. The possibility of having providers with potentially winning bids linked to edge nodes unavailable to host users is not excluded. This case stems from the condition in which the available resources on the edge node, i.e., Eq.* (14)*, are insufficient to accept the user requiring the minimum cost. Furthermore, the case where a provider offers valuable bids, but it is never the best, may occur.*

4.3. Stability Analysis

The algorithm subtends dependencies among the preferences of players [15]. For matching games with externalities no algorithm exists that guarantees a stable outcome. In order to discuss the stability of the matching stemming by the proposed framework, we consider an S2ES stability definition, modifying the original version of [38] to obtain a concept of stability useful in our problem. Specifically, we define outcome matching as stable when a swap is allowed if an improvement to at least one user and one edge node involved is provided, and the remaining players participating in the swap do not worsen. Formally, this is expressed by the following

Definition 2. Let $\mathcal{Z}$ be the outcome matching of the developed algorithm, and let $\mathcal{Z}(c)$ be the edge node matched with the user c in the matching $\mathcal{Z}$. The outcome matching $\mathcal{Z}$ is an S2ES matching *if* there not exists a pair of users (c_1, c_2) s.t.:

1. $H_{c_1}(\mathcal{Z}(c_2)) \geq H_{c_1}(\mathcal{Z}(c_1))$ and
2. $H_{c_2}(\mathcal{Z}(c_1)) \geq H_{c_2}(\mathcal{Z}(c_2))$ and
3. $E_{\mathcal{Z}(c_1)}(c_2) \geq E_{\mathcal{Z}(c_1)}(c_1)$ and

4. $E_{\mathcal{Z}(c_2)}(c_1) \geq E_{\mathcal{Z}(c_2)}(c_2)$ and
5. $\exists \psi \in \{c_1, c_2\}$ s.t. at least one of the conditions $1) - 2)$ is strictly verified and
6. $\exists \phi \in \{\mathcal{Z}(c_1), \mathcal{Z}(c_2)\}$ s.t. at least one of the conditions $3) - 4)$ is strictly verified.

Lemma 2. The outcome matching is a S2ES configuration.

Proof. To prove the stability of the proposed matching algorithm, we proceed by *reductio ad absurdum*, assuming the existence of a pair of users (c_1, c_2) for which conditions $1)-2)$ of Definition 2 hold. Furthermore, let c_1 and c_2 be such that $\mathcal{Z}(c_1) = a_1$ and $\mathcal{Z}(c_2) = a_2$, respectively. This necessarily implies that:

$$H_{c_1}(a_2) \geq H_{c_1}(a_1), \quad \text{and} \quad H_{c_2}(a_1) \geq H_{c_2}(a_2). \tag{18}$$

Given the satisfaction of condition 5) of Definition 2, we observe that the proposed assignment policy does not incorporate any discard strategy. Consequently, the number of available resources at edge nodes cannot increase during the assignment process. This implies that the preference list of each matched user remains unchanged after its assignment. Since, upon assigning a generic user to an edge node, the available resources on that node do not increase, it follows that at most:

$$H_{c_1}(a_1) = H_{c_1}(a_2), \quad \text{and} \quad H_{c_2}(a_2) = H_{c_2}(a_1). \tag{19}$$

As a direct consequence, condition 5) is not satisfied. Therefore, even if condition 6) holds, the proposed matching game reaches a configuration that satisfies the S2ES property. □

Although edge nodes can only select their preferred user from the set of received proposals, the algorithm ensures that if a user c is assigned to an edge node a, which was ranked in position i in its preference list, then all edge nodes ranked higher in that list have been assigned to other users with whom they achieve a higher gain than they would have obtained with c. This guarantees that each user is assigned to the best possible edge node under the given constraints. Such a result can be formally stated as follows.

Lemma 3. Every user is matched with the best possible edge node according to their preferences.

Proof. We prove the result by contradiction. Assume that there exists at least one user who is not assigned to their best possible edge node. Let c be the first user to receive a rejection from an edge node, and let $\mathcal{Z}(c)$ be the edge node assigned to c by the matching algorithm. This implies that there exists a more preferred edge node $\mathcal{Z}_{better}$ from which c was previously rejected. By construction, c initially preferred $\mathcal{Z}_{better}$ and proposed an assignment to it. However, $\mathcal{Z}_{better}$ rejected c in favor of another user c', whom $\mathcal{Z}_{better}$ preferred and who also proposed to $\mathcal{Z}_{better}$. Therefore, when receiving proposals from both c and c', the edge node $\mathcal{Z}_{better}$ accepted c' and rejected c. Since $\mathcal{Z}_{better}$ prefers c' to c, and c prefers $\mathcal{Z}_{better}$ to $\mathcal{Z}(c)$, the only way to match c with $\mathcal{Z}_{better}$ in a stable configuration is if there exists an even more preferred edge node $\mathcal{Z}'_{better}$ that previously rejected c', thereby inducing c' to propose to $\mathcal{Z}_{better}$. However, this contradicts our initial assumption that c is the first user to be rejected by an edge node. Therefore, our assumption must be false, and every user must be matched with their best possible edge node according to the algorithm's stability constraints. □

4.4. Complexity Analysis

To assess the computational complexity of the proposed framework, we consider the worst-case scenario in which every edge node can potentially serve any user. Each user must rank all edge nodes in the set $\mathcal{A}$ according to the preference relation defined in Eq. (13). For each user, this sorting procedure requires $O(A \cdot \log A)$ operations, leading to an overall complexity of $O(C \cdot A \cdot \log A)$ across all users in $\mathcal{C}$. Similarly, each edge node constructs its preference list over the users in $\mathcal{C}$, resulting in a complexity of $O(A \cdot C \cdot \log C)$. The construction of these preference lists involves running a VCG auction for each service required by the workflows. In the worst case, we assume that every provider offers its CDF to all services composing each workflow, and that every workflow requires all services in $\mathcal{S}$. This results in an additional complexity term of

$$O\big(A \cdot C \cdot (S \cdot P + \log C)\big),$$

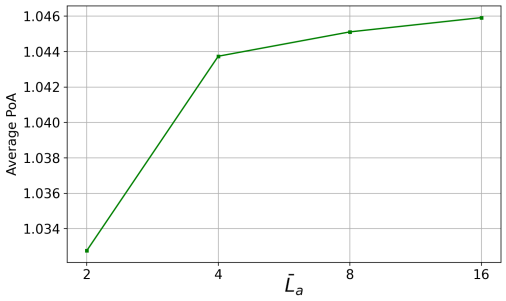


Figure 1: Average Price of Anarchy as the number of mean edge node resources increases

yielding a total complexity of

$$O(C \cdot A \cdot \log A) + O\big(A \cdot C \cdot (S \cdot P + \log C)\big).$$

Since typically $A \ll C$ and the algorithm terminates within C iterations, the overall computational complexity can be expressed as:

$$O\big(A \cdot C^2 \cdot (S \cdot P + \log C)\big). \tag{20}$$

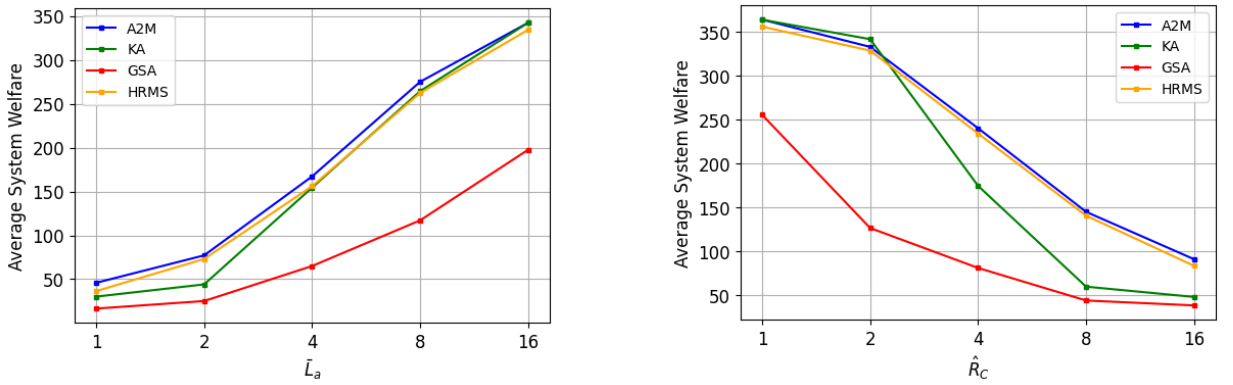
5. Performance Analysis

In this section, we test the applicability and effectiveness of the proposed Auction-to-Matching (A2M) strategy. We first assess that casting GEN distributions into uniform CDF bids causes a negligible degradation of system performance. In so doing, it is possible to assess whether the information approximated still catches the dynamics of the problem without causing excessive losses in terms of the system welfare. Then, alternative algorithms are implemented to provide performance evaluation, also comparing the A2M with centralized and user-oriented approaches. Finally, the influence of unexpected delays and competitiveness among providers is explored. Summarizing, the experimentation objectives are captured by the following research questions:

- **Q1.** How does the approach perform in terms of system welfare in comparison to a strategy assuming full knowledge about providers' CDFs?
- **Q2.** How does the combinatorial space dimension affect system welfare and the probability that a user cannot be served?
- **Q3.** How do unexpected delays and competitiveness among providers affect system welfare?

In this experiment, the following setting is considered. The cost R_c associated to user requests, and the capacity L_a of each edge node a are uniformly selected over $[0, 3.0]$, $[2.0, 20.0]$, respectively, and terms $\bar{L}_a$ and $\bar{R}_c$ are used to denote the mean value of the edge node resources and the mean value of request costs, respectively. Without loss of generality, the price b_c offered by user c is set to 20.0 which is reasonable to illustrate the system dynamics. We assume that constraints about two parameters a and b of Eq. (9) are set by the edge node by requiring that submitted uniform distributions have a specified value χ of the ratio $\frac{b}{a}$, which in the end amounts to setting a constraint on the coefficient of variation. According to this, parameters a and b are determined as the solution of the following system of equations

$$\begin{cases} \frac{1}{a(\chi-1)} \int_a^b F_X(x)dx = \frac{1}{2} \\ b = \chi \cdot a \end{cases} \tag{21}$$



(a) System welfare as the number of mean edge node resources increases

(b) System welfare as the mean value of request costs increases

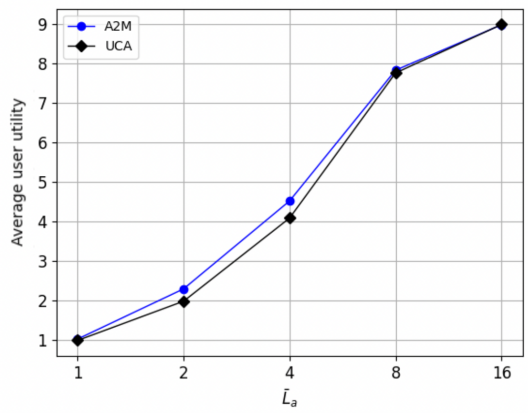
Figure 2: System welfare under different system parameters: (a) impact of the average edge node resources, and (b) impact of the mean request cost

We consider 3 distinct workflow topologies, each of which is built by combining 8 individual sub-tasks orchestrated in *sequential*, *parallel*, and *mixed* topology. The mixed topology incorporates elements of both sequential and parallel topology, enabling services to be combined in both chains of services and patterns where race conditions among services are present. In particular, the mixed topology is the parallel of 2 sequences, where each sequence is in turn the parallel of 2 services. Without loss of generality, the SLO F_c expressed by each user c can be assumed as a uniform CDF whose support is arbitrarily chosen to fit the expected workflow e2e time $E[t_E]$ and transitions are supported over $[2.5, 7.5]$.

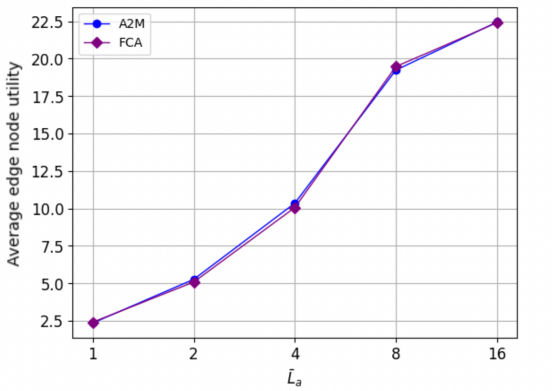
For each service s, bids are generated as follows. First, for each provider p bidding for the service s, a GEN distribution X_p representing the delivery duration guaranteed by p is randomly selected from a set of 100 histograms. For each run of simulations, histograms are generated by sampling the WS-Dream dataset [39]. The WS-Dream dataset provides response-time measurements collected from a large set of web services and geographically distributed users. Although originally released in 2008, it remains one of the most widely adopted public benchmarks for evaluating Web-service QoS and latency prediction, owing to its scale, accessibility, and standardized structure. Its widespread use in recent studies motivates our choice of employing it to derive the response-time histograms used in our experiments [40, 41, 42]. WS-Dream response times are labeled with a service type. To generate a histogram, we select a service type and we collect all the related response times. Before generating the histogram, we regularized the collected data by exploiting the inter-quartile range rule, also known as Tukey's rule of thumb [43]. Finally, regularized samples are collected in histograms with 64 bins between the lower and the highest values of the samples. Then, each provider bid is obtained by casting the selected histogram into the uniform approximation defined in Eq. (21). The best provider and second-best provider are determined by evaluating the pairwise-comparison dominance on the obtained uniform bids as in Section 3.1. Finally, the stochastic behavior of the considered service s is characterized as the GEN distribution $\tilde{X}_\star$ of the best provider $p^\star$, which is rewarded according to the CDF F_Y of the second best provider uniform bid. Hence, the reward of the winning provider is calculated as $\int_a^b (1 - F_{Y^{uni(a,b)}}(x) \cdot f_{\tilde{X}_\star}(x))dx$.

As all actual sub-task durations have been characterized, the completion time t_E of the workflow is evaluated, and the reward of the edge node is computed as $\int_a^b (1 - F_c(x) \cdot f_{t_E}(x))dx$.

The approach has been implemented in Java, using the SIRIO Library [44] of the ORIS tool [36] to represent and manipulate CDFs. The implementation exploits also the Eulero library [37] to efficiently compute the pdf f_{t_E} of the e2e time t_E of the workflow of each user c to derive the expected reward of edge node a serving user c as $E[\mathcal{U}_{c,a}]$. The experiments illustrated in the following subsections are performed using a single core of an Intel Xeon Gold 5120 CPU (2.20 GHz) equipped with 32 GB of RAM.



(a) User utility as the number of mean edge node resources increases



(b) Edge node utility as the number of mean edge node resources increases

Figure 3: Impact of increasing average edge node resources on user-side and edge-side utilities

5.1. Analysis of the Price of Anarchy

To answer question Q1, we implement a variant of the proposed approach, where full knowledge about provider CDFs is exploited. In principle, cast GEN CDFs to the shape of uniform distribution might break the selection of the actual best and second best bids among all the proposals. To manage this complexity, an oriented graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is built to map the mutual pairwise-comparison dominance relations existing among provider CDFs. The set $\mathcal{V}$ is the set of vertices, here represented as provider CDFs. Two vertices X and Y are connected by an oriented edge (from X to Y), whose set is $\mathcal{E}$, when $X \preceq Y$. Then, the vertex $X^\star$ with the highest outbound degree (i.e., the highest number of outbound edges) is identified as the first best bid, while the second best bid is the vertex connected to $X^\star$ through an inbound edge, having the highest outbound degree.

To measure the performance degradation due to the uniform cast of the actual provider CDFs, we compute a Price of Anarchy (PoA) [45, 46], here defined as the ratio between the expected system welfare achieved by the full-knowledge approach and that achieved by the proposed approach. Specifically, we consider $C = 20$ users, $\bar{L}_a$ uniformly distributed within $[2, 16]$, 1 workflow with random topology (among sequential, parallel, and mixed) for each user. For each value of $\bar{L}_a$ considered, we generate 1000 instances of the problem, deriving the average PoA.

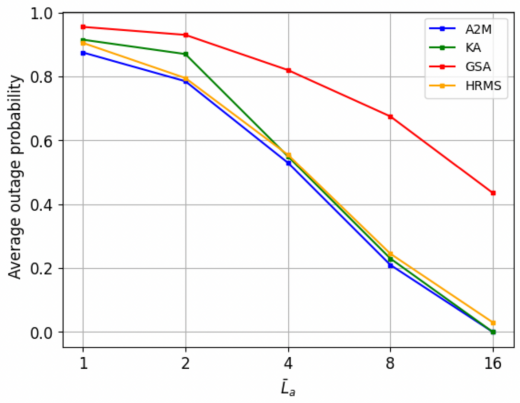
Results, reported in Fig. 1, show that the average PoA increases with $\bar{L}_a$, which, in turn, determines the dimension of the combinatorial space of the problem. This means that the proposed approach suffers only slightly from the lack of knowledge due to the introduction of CDF uniform approximations when the combinatorial possibilities grow. Results confirm the validity of the proposed low-complexity approach in achieving satisfactory performance while adopting the lightweight A2M approach.

5.2. Analysis of System Welfare, Outage Probability, and User Utility

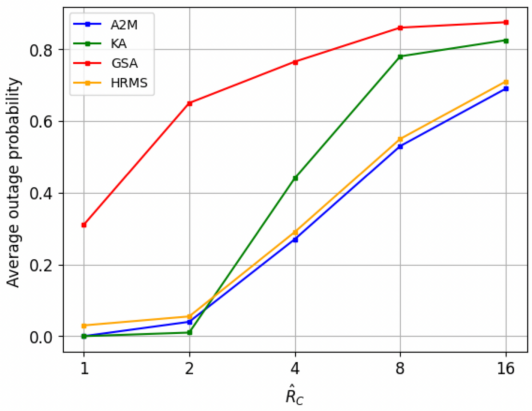
To answer question Q2, we implemented the following additional decision-making schemes from the literature

- *Gale-Shapley Algorithm* (GSA): the GSA is applied in its traditional form, without updating preference lists after each assignment [15];
- *Enhanced Kolkata Algorithm* (KA):

  To explicitly quantify the effect of separating offloading from service selection, we design a deliberately decoupled baseline inspired by the Kolkata repeated game [47]. In this configuration, users construct their preference lists following Eq. (13), but edge nodes admit users in a fully service-agnostic manner:



(a) Outage probability as the number of mean edge node resources increases



(b) Outage probability as the mean value of request costs increases

Figure 4: Outage probability under varying system conditions, highlighting the impact of resource availability and request cost intensity

each edge node independently and randomly selects among the proposing users, without exploiting any information on service types or provider performance. This mechanism intentionally breaks the coupling that characterizes the proposed A2M scheme. As a consequence, the offloading decision (which user attaches to which edge) and the subsequent service selection evolves in isolation, with no feedback loop between them. The KA mechanism updates user preferences every three iterations, providing an adaptive yet computationally light comparison baseline.

- *Heterogeneous Resource Management Scheme (HRMS)*:

  To enable a fair comparison with recent DAG-based offloading strategies, we adapt the heterogeneous resource–management scheme proposed in [48] to the context of our workflow-centric model. In doing so, we redesign the HRMS procedure so that it exploits the same information available to A2M and follows an analogous decision logic, while preserving its original scalar utility formulation. This alignment ensures that the two schemes rely on conceptually equivalent inputs and use them in a comparable manner, which explains why their performance remains largely similar, with only minor differences arising from the specific form of the HRMS metric. More specifically, for every pair (c, a), we compute a scalar score defined as

$$M_{c,a} = \omega_1(\mathcal{L}_a - R_c) + (1-\omega_1)\left(\int_{x=0}^{\infty} f_{\tilde{X}_\star}(x)(1-F_c(x))dx - \frac{1}{2}\right), \tag{22}$$

  where the first term captures the residual capacity exposed by edge node a, and the second one quantifies the expected reward based on the sub-task completion distributions. In our implementation we set $\omega_1 = 0.5$, reflecting an equal weighing between load-awareness and service-performance awareness, in line with the multi-objective balancing performed by HRMS in [48]. For each user c, the adapted HRMS baseline selects the edge node

$$a_c^\star = \arg\max_{a\in\mathcal{A}} M_{c,a}, \tag{23}$$

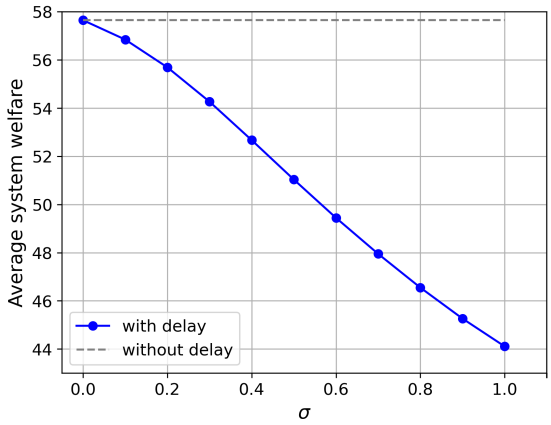
  thus emulating the frontier-based evaluation logic used in the original algorithm, where candidate placements are scored through a weighed combination of structural and performance indicators. This yields a centralized, optimization-oriented reference scheme that leverages the same information primitives available to A2M, while remaining agnostic to the game-theoretic interactions among users and providers.

- *User-Centric Algorithm* (UCA): users build preference lists with the aim of maximizing their individual utility, expressed in terms of responsiveness in receiving service, thus defined as $1 - F_c(t_E)$. Edge node preference lists are built in accordance with (15).
- *Fully Centralized Auction* (FCA): users offer to a centralized auctioneer individual bids defined as $1 - F_c(t_E)$. Then, the assignment is provided considering the pair user-edge node which maximizes Eq. (16).

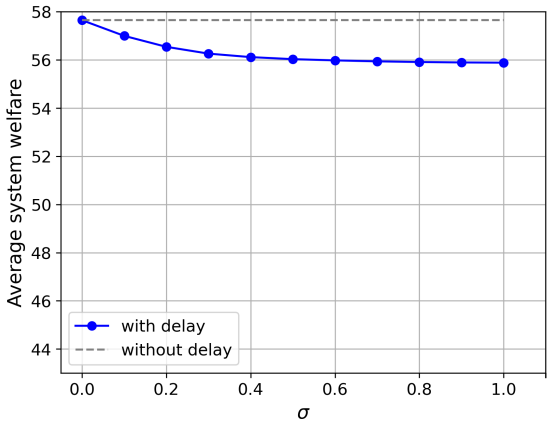
Also in this case, results are obtained averaging 1000 instances of the problem. Recalling that $\bar{L}_a$ and $\bar{R}_c$ are the mean value of the edge node resources and the mean value of request costs, we can study the behavior of the system as $\bar{L}_a$ and $\bar{R}_c$ change. In so doing, we explore the ability of the proposed strategy to find suitable solutions even when the combinatorial space of solutions increases. In this reference, the average system welfare as a function of $\bar{L}_a$ and $\bar{R}_c$ is plotted in Fig. 2a and Fig. 2b, respectively. In Fig. 2a, the A2M curve increases as the $\bar{L}_a$ grows, i.e., when the edge node resources become coarser. In fact, the bigger the values of $\bar{L}_a$, the better the availability of edge nodes in hosting users which, in turn, raises the system welfare. From Fig. 2a follows that A2M guarantees a better resource exploitation, compared to the alternatives, in presence of scarcity of edge node capacity, i.e., low values of $\bar{L}_a$. When $\bar{L}_a$ increases, the system load decreases, inducing a less congested scenario. In such a situation, the gap between KA e A2M tends to be neglected, since the resources are abundant. The advantages of the proposed A2M are also evident by setting in Fig. 2a an average outage probability target equal to 0.8. In this case, with the A2M approach, target satisfaction can be achieved with mean edge resources $\bar{L}_a = 1.8$, against the KA and the GSA that require $\bar{L}_a = 2.6$ and $\bar{L}_a = 4.2$, respectively. This implies relevant benefits for the provider infrastructure due to the effective resource exploitation reached adopting the A2M. On the contrary, in Fig. 2b, the system welfare drops when $\bar{R}_c$ increases, i.e., when the cost associated with user requests becomes heavier. In fact, $\bar{R}_c$ rules the impact of users in being allocated on edge nodes. When $\bar{R}_c$ grows, the number of users allocated decreases, resulting in lower system welfare. Once again, A2M reaches a better optimization of the edge node capacities when the system is congested. Moreover, the importance of the nested auction serving the matching game is evident in both Fig. 2a and Fig. 2b when the resources are scarce, i.e., low values of $\bar{L}_a$ and high values of $\bar{R}_c$, respectively. Both Fig. 2a and Fig. 2b show remarkable performance improvements of A2M on GSA and KA. HRMS and A2M display similar performance because they are based on the same set of information and both address the offloading and service–selection problems in a joint manner. The only additional element used by A2M is the term $(b_c - R_c)$, which appears in its preference lists. This extra component introduces a more refined optimization step compared to the mixed criterion employed in HRMS.

In Fig. 3a and Fig. 3b the behavior of a user utility and edge node utility are illustrated, comparing A2M with UCA and FCA, respectively. Fig. 3a evidences that the proposed A2M approach does not penalize the individual interest of users. The reason is that in the A2M the user interests are indirectly safeguarded by the edge node preference list metric defined in Eq.(16). Accordingly, the edge node increases its utility when the user receives the service in time. Fig. 3b depicts the trend of the edge node utility when the FCA is adopted. As it is evident from the Figure, the A2M reaches the same performance as the centralized approach with a much lower complexity. This is a direct consequence of the metric exploited in the A2M during the matching game, where not exclusively local performance metrics are involved in the preference list process construction.

Fig. 4a illustrates the trend of the outage probability, defined as the probability that a user cannot be served, as a function of $\bar{L}_a$, keeping $R_c = 2$ for each user. Increasing the available edge node resources, users have a greater probability to be accepted by edge nodes, implying a lower outage probability. Fig. 4b shows the outage probability behavior when $\bar{R}_c$ varies and $L_a = 10$ for all edge nodes. Since edge node capacity is fixed, increasing the values of $\bar{R}_c$, edge node acceptance capability suffers significant limitations, reaching high outage probability values. In this case as well, A2M and HRMS confirm comparable performance, with a slight advantage in favor of A2M.



(a) Average edge node utility as the slowing factor σ increases for all services (balanced delay)



(b) Average edge node utility as the slowing factor σ increases for a single service (unbalanced delay)

Figure 5: Impact of the slowing factor σ on average edge node utility under balanced and unbalanced delay conditions

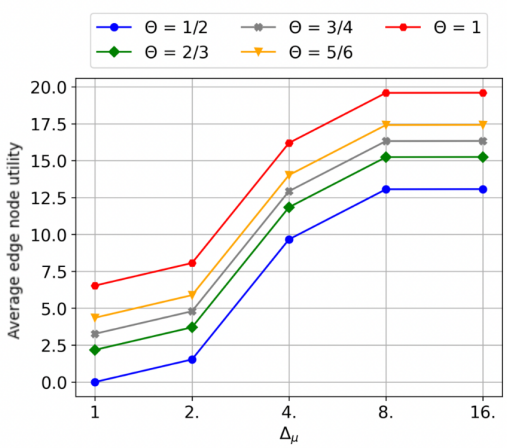
5.3. Analysis Under Unexpected Delays

To answer question Q3, actual service times are mutated with respect to the expected distribution expressed in the bidding phase. To this end, since the results of the analysis are invariant with respect to the number of edge nodes and users considered, we can safely assume $A = 1$ and $C = 1$, and a sequential workflow which is the topology that maximally emphasizes the correlation between the e2e delay and the individual service delay. Two different kinds of delays are tested: in the former, all services of the workflow are subject to a delay (*balanced delay*); in the latter, all the delay is concentrated on a single service (*unbalanced delay*). In the nominal case without delay, for each service s supplied by provider p, the CDF $\tilde{F}_s$ of the actual completion time corresponds to the claimed CDF. In the case of balanced delay, each service s supplied by provider p is delayed by increasing the support bounds (and so the expected value) by a factor $\sigma \in \{0.1, 0.2, \ldots, 1\}$, e.g., if $\sigma = 0.1$, then the support bounds and the expected service time are increased by 10% with respect to the nominal case. Conversely, in the case of unbalanced delay, the variation is applied only to a single service s of the workflow, by increasing the support bounds and the expected value of a factor of $\tilde{F}_s$ by σ with $\sigma \in \{8 \cdot 0.1, 8 \cdot 0.2, \ldots, 8 \cdot 1\}$, where 8 is the number of services of the workflow.

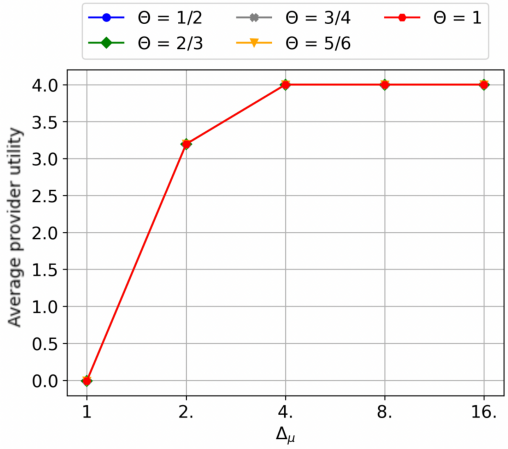
For each value of σ, 200 instances of the problem are generated, deriving the average edge node utility, as illustrated in Figs. 5a and 5b, for the cases of balanced and unbalanced delay, respectively. As expected, the average edge node utility decreases as the slowing factor σ increases, since the expected actual completion time of the workflow increases. Note that, as σ increases, the reduction in the average utility is larger in the balanced case, when an expected delay of $100 \cdot \sigma\,\%$ affects each of the 8 services, rather than in the unbalanced case, when an expected delay of $8 \cdot 100 \cdot \sigma\,\%$ affects a single service, given that the expected delay in the e2e workflow time is larger in the first case. Note that for both the considered cases, the workflow completion time is equally delayed, as a consequence of using uniform distributions to characterize sub-tasks of the workflow. Since the completion time is identically delayed for the tested cases, the average utility of the edge node (first addend in Eq. (3)) is the same in both cases. Instead, as the delay amount increases, the cumulative utility of providers (second addend in Eq. (3)) is more penalized when many edge nodes result in being delayed. Consequently, the utility of the system tends to decrease more slowly in the unbalanced delay case than in the balanced case. This suggests that the presence of very reliable providers succeeds in shielding the presence of an unreliable provider, more than many equally reliable providers are able to.

5.4. Analysis Under Variable Competitiveness

To further answer Q3, the impact of the competitiveness among providers on the edge node utility and the provider utility is evaluated by varying provider support distributions according to a scaling factor Δ_μ.



(a) Average edge node utility as a function of Δ_μ for different values of Θ



(b) Average provider utility as a function of Δ_μ for different values of Θ

Figure 6: Impact of the parameter Δ_μ on edge node and provider utilities under different values of the control threshold Θ

Factor Δ_μ rules the earliness of the expected value and the width of the support of the uniform CDF of the best bid with respect to the expected value and the width of the support of the uniform CDF of the second best bid, identified by the VCG auction banned for each service s. Without loss of generality, we consider the second best bid as a fixed uniform distribution with expected value $\mu_{\text{2nd}} = 10$ and support $[7.5, 12.5]$. The best bid is generated as a uniform distribution having expected value $\mu_{\text{1st}} = \frac{3\mu_{\text{2nd}}}{2\Delta_\mu}$ and support $\left[\frac{2\mu_{\text{1st}}+\log_2\Delta_\mu-\mu_{\text{2nd}}/2}{2}, \frac{2\mu_{\text{1st}}-\log_2\Delta_\mu+\mu_{\text{2nd}}/2}{2}\right]$, with $\Delta_\mu \in \{1, 2, 4, 8, 16\}$. In this experiment, we still consider $A = 1$ and $C = 1$, where the user requests one workflow with mixed topology. For each value of Δ_μ, 200 instances of the problem are generated, deriving the average edge node welfare and the average provider utility, as illustrated in Figs. 6a and 6b. As Δ_μ increases, the average edge node utility in Fig. 6a increases as $(1 - F_c(t_E))$ grows. From a certain value of Δ_μ on, $(1 - F_c(t_E))$ becomes equal to 1 and thus the average edge node utility remains almost constant. To analyze average edge node utility under different market settings, experiments are repeated with $\Theta \in \{\frac{1}{2}, \frac{2}{3}, \frac{3}{4}, \frac{5}{6}, 1\}$. For each value of Δ_μ, the greatest value of the average edge node utility (Fig. 6a) is obtained with $\Theta = 1$, which represents the best-case condition where edge nodes receive the highest possible payment from users according to Eq. (2). According to this, as the value of Θ decreases, also the average edge node utility decreases.

Similarly, the average provider utility in Fig. 6b increases with Δ_μ, up to a certain value after which it remains constant, which comprises a direct consequence of the VCG mechanism.

6. Conclusion

This paper addressed the joint offloading and service selection problem in a resource-constrained edge computing environment, where sub-task dependencies play a crucial role in determining service execution effectiveness. Given the emergence of computational intensive applications such as those AI-driven and their increasingly complex computational needs, optimizing processing resources is critical to ensuring low-latency inference and efficient task execution.

To tackle this challenge, we proposed a stochastic framework that integrates a matching game with externalities to efficiently assign users to edge nodes, while a low-complexity CDF-driven VCG auction selects the most suitable providers for executing sub-tasks. Unlike traditional approaches that assume static execution times, our framework introduces probabilistic modeling, where user SLOs are expressed as CDFs of e2e workflow completion times, and provider bids using CDFs of sub-task completion times, approximated through uniform distributions.

Since we assume a high-speed, well-designed communication infrastructure, we disregard transmission delays and focus entirely on computational challenges, making our approach particularly relevant for AI-driven services requiring intensive processing. The proposed A2M mechanism was evaluated through extensive simulations, demonstrating its effectiveness in meeting user-defined SLOs, improving resource utilization, and minimizing service latency. Compared to benchmark methods, A2M achieves superior performance in terms of both efficiency and scalability, making it a promising solution for large-scale, distributed AI inference in edge-to-cloud architectures. Future research could extend this framework by exploring adaptive learning-based allocation policies, and enhancing collaborative federated AI execution models. Additionally, investigating dynamic pricing strategies in auction-based mechanisms could further improve the fairness and sustainability of edge service provisioning.

References

[1] Z. Zhou, X. Chen, E. Li, L. Zeng, K. Luo, J. Zhang, Edge intelligence: Paving the last mile of artificial intelligence with edge computing, Proceedings of the IEEE 107 (8) (2019) 1738–1762. doi:10.1109/JPROC.2019.2918951.

[2] H. Huang, Q. Ye, Y. Zhou, 6G-empowered offloading for realtime applications in multi-access edge computing, IEEE Transactions on Network Science and Engineering 10 (3) (2023) 1311–1325. doi:10.1109/TNSE.2022.3188921.

[3] D. Liu, A. Hafid, L. Khoukhi, Workload balancing in mobile edge computing for internet of things: A population game approach, IEEE Transactions on Network Science and Engineering 9 (3) (2022) 1726–1739. doi:10.1109/TNSE.2022.3150755.

[4] M. Straesser, S. Geissler, S. Lange, L. K. Schumann, T. Hossfeld, S. Kounev, Trust your local scaler: A continuous, decentralized approach to autoscaling, Performance Evaluation 167 (2025) 102452. doi:10.1016/j.peva.2024.102452.

[5] A. Koley, C. Singh, Optimal resource management for multi-access edge computing without using cross-layer communication, Performance Evaluation 166 (2024) 102445. doi:10.1016/j.peva.2024.102445.

[6] X. Zhang, T. Lin, C.-K. Lin, Z. Chen, H. Cheng, Computational task offloading algorithm based on deep reinforcement learning and multi-task dependency, Theoretical Computer Science 993 (2024) 114462. doi:10.1016/j.tcs.2024.114462.

[7] K. Peng, P. Liu, M. Bilal, X. Xu, E. Prezioso, Mobility and privacy-aware offloading of ar applications for healthcare cyber-physical systems in edge computing, IEEE Transactions on Network Science and Engineering 10 (5) (2023) 2662–2673. doi:10.1109/TNSE.2022.3185092.

[8] Q. Shen, B.-J. Hu, E. Xia, Dependency-aware task offloading and service caching in vehicular edge computing, IEEE Transactions on Vehicular Technology 71 (12) (2022) 13182–13197. doi:10.1109/TVT.2022.3196544.

[9] G. Zhao, H. Xu, Y. Zhao, C. Qiao, L. Huang, Offloading tasks with dependency and service caching in mobile edge computing, IEEE Transactions on Parallel and Distributed Systems 32 (11) (2021) 2777–2792. doi:10.1109/TPDS.2021.3076687.

[10] W. Zhao, R. Chellappa, P. J. Phillips, A. Rosenfeld, Face recognition: A literature survey, ACM Comput. Surv. 35 (4) (2003) 399–458. doi:10.1145/954339.954342.

[11] D. K. Barry, D. Dick, Chapter 3 - web services and service-oriented architectures, in: D. K. Barry, D. Dick (Eds.), Web Services, Service-Oriented Architectures, and Cloud Computing (Second Edition), The Savvy Manager's Guides, Morgan Kaufmann, Boston, 2013, pp. 15–33. doi:10.1016/B978-0-12-398357-2.00003-8.

[12] M. P. Papazoglou, W. J. van den Heuvel, Service oriented architectures: approaches, technologies and research issues, The VLDB Journal 16 (3) (2007) 389–415. doi:10.1007/s00778-007-0044-3.

[13] F. Sheikholeslami, N. Jafari Navimipour, Auction-based resource allocation mechanisms in the cloud environments: A review of the literature and reflection on future challenges, Concurrency and Computation: Practice and Experience (2018). doi:10.1002/cpe.4456.

[14] G. Baranwal, D. P. Vidyarthi, A truthful and fair multi-attribute combinatorial reverse auction for resource procurement in cloud computing, IEEE Transactions on Services Computing (2019). doi:10.1109/TSC.2016.2632719.

[15] D. Manlove, Algorithmics of matching under preferences, Vol. 2, World Scientific, 2013.

[16] G. Gao, M. Xiao, J. Wu, H. Huang, S. Wang, G. Chen, Auction-based VM allocation for deadline-sensitive tasks in distributed edge cloud, IEEE Transactions on Services Computing (2021). doi:10.1109/TSC.2019.2902549.

[17] V. Vazirani, Approximation Algorithms, Springer Berlin Heidelberg, 2013.

[18] M. Moghaddam, J. Davis, Simultaneous service selection for multiple composite service requests: A combinatorial auction approach, Decision Support Systems 120 (03 2019). doi:10.1016/j.dss.2019.03.005.

[19] Y. Fan, X. Cai, W. Yue, J. Zheng, C. Li, A deep reinforcement learning approach for dependency-aware task offloading in cooperative vehicular networks, in: 2023 IEEE 34th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC), 2023, pp. 1–6. doi:10.1109/PIMRC56721.2023.10294053.

[20] J. Ma, R. Yao, B. Zhang, Z. Wang, Y. Yan, Data-driven flexibility capability modeling of internet data center considering task dependency, IEEE Internet of Things Journal 11 (14) (2024) 24538–24550. doi:10.1109/JIOT.2024.3395837.

[21] K. Mishra, G. N. V. Rajareddy, U. Ghugar, G. S. Chhabra, A. H. Gandomi, A collaborative computation and offloading for compute-intensive and latency-sensitive dependency-aware tasks in dew-enabled vehicular fog computing: A federated deep q-learning approach, IEEE Transactions on Network and Service Management 20 (4) (2023) 4600–4614. doi:10.1109/TNSM.2023.3282795.

[22] X. Chen, J. Cao, Y. Sahni, S. Jiang, Z. Liang, Dynamic task offloading in edge computing based on dependency-aware reinforcement learning, IEEE Transactions on Cloud Computing 12 (2) (2024) 594–608. doi:10.1109/TCC.2024.3381646.

[23] X. Zhou, S. Ge, P. Liu, T. Qiu, DAG-based dependent tasks offloading in MEC-enabled IoT with soft cooperation, IEEE Transactions on Mobile Computing 23 (6) (2024) 6908–6920. doi:10.1109/TMC.2023.3328333.
[24] Y. Bian, Y. Sun, M. Zhai, W. Wu, Z. Wang, J. Zeng, Dependency-aware task scheduling and offloading scheme based on graph neural network for MEC-assisted network, in: 2023 IEEE/CIC International Conference on Communications in China (ICCC Workshops), 2023, pp. 1–6. doi:10.1109/ICCCWorkshops57813.2023.10233785.
[25] T. G. Chetan, M. Jenamani, S. P. Sarmah, Two-stage multi-attribute auction mechanism for price discovery and winner determination, IEEE Transactions on Engineering Management (2019). doi:10.1109/TEM.2018.2810510.
[26] G. Baranwal, D. P. Vidyarthi, A fair multi-attribute combinatorial double auction model for resource allocation in cloud computing, Journal of Systems and Software 108 (2015). doi:10.1016/j.jss.2015.06.025.
[27] A. I. Middya, B. Ray, S. Roy, Auction based resource allocation mechanism in federated cloud environment: Tara, IEEE Transactions on Services Computing (2019). doi:10.1109/TSC.2019.2952772.
[28] X. Wang, Y. Sui, J. Wang, C. Yuen, W. Wu, A distributed truthful auction mechanism for task allocation in mobile cloud computing, IEEE Transactions on Services Computing 14 (3) (2021). doi:10.1109/TSC.2018.2818147.
[29] W. Shi, L. Zhang, C. Wu, Z. Li, F. C. Lau, An online auction framework for dynamic resource provisioning in cloud computing, SIGMETRICS '14, Association for Computing Machinery, New York, NY, USA, 2014. doi:10.1145/2591971.2591980.
[30] C. Jiang, Z. Luo, L. Gao, J. Li, A truthful incentive mechanism for movement-aware task offloading in crowdsourced mobile edge computing systems, IEEE Internet of Things Journal 11 (10) (2024) 18292–18305. doi:10.1109/JIOT.2024.3362406.
[31] J. Huang, S. Li, L. Yang, J. Si, X. Ma, S. Wang, Multiparticipant double auction for resource allocation and pricing in edge computing, IEEE Internet of Things Journal 11 (8) (2024) 14007–14016. doi:10.1109/JIOT.2023.3339655.
[32] A. Papakonstantinou, P. Bogetoft, Multi-dimensional procurement auction under uncertain and asymmetric information, European Journal of Operational Research 258 (3) (2017). doi:10.1016/j.ejor.2016.09.060.
[33] S. Bernardi, J. Campos, J. Merseguer, Timing-failure risk assessment of UML design using time Petri net bound techniques, IEEE Transactions on Industrial Informatics 7 (1) (2010) 90–104.
[34] J. Bengtsson, K. Larsen, F. Larsson, P. Pettersson, W. Yi, Uppaal: a tool suite for automatic verification of real-time systems, in: Proc.of the DIMACS SYCON Workshop on Verification and Control, Springer-Verlag, Berlin, Heidelberg, 1996.
[35] J. F. Allen, Maintaining knowledge about temporal intervals, Communications of the ACM 26 (11) (1983).
[36] M. Paolieri, M. Biagi, L. Carnevali, E. Vicario, The ORIS tool: Quantitative evaluation of non-Markovian systems, IEEE Transactions on Software Engineering 47 (6) (2021). doi:10.1109/TSE.2019.2917202.
[37] L. Carnevali, M. Paolieri, R. Reali, E. Vicario, Compositional safe approximation of response time probability density function of complex workflows, ACM Transactions on Modeling and Computer Simulation 33 (4) (2023) 16:1–16:26. doi:10.1145/3591205.
[38] E. Bodine-Baron, C. Lee, A. Chong, B. Hassibi, A. Wierman, Peer effects and stability in matching markets, Vol. 6982, 2011. doi:10.1007/978-3-642-24829-0_12.
[39] Z. Zheng, M. R. Lyu, Ws-dream: A distributed reliability assessment mechanism for web services, in: 2008 IEEE Int. Conf. on Dependable Systems and Networks (DSN), IEEE, 2008.
[40] L. Kong, X. Hu, L. Qi, X. Xu, Y. Zhang, L. Yao, X. Zhang, Deep Learning to Hash for Time-Aware QoS Prediction Based on VQ-VAE , IEEE Transactions on Services Computing 18 (05) (2025) 2726–2739.
[41] G. Zou, S. Lin, S. Wu, S. Hu, S. Yang, Y. Gan, B. Zhang, Y. Chen, Combining personalized federated hypernetworks and shared residual learning for distributed QoS prediction 20 (3) (2025). doi:10.1145/3709141.
[42] S. Hu, G. Zou, B. Zhang, S. Wu, S. Lin, Y. Gan, Y. Chen, GACL: Graph attention collaborative learning for temporal QoS prediction, IEEE Transactions on Network and Service Management 22 (4) (2025) 3388–3402. doi:10.1109/TNSM.2025.3570464.
[43] J. W. Tukey, et al., Exploratory data analysis, Vol. 2, Reading, MA, 1977.
[44] Sirio library, `https://github.com/oris-tool/sirio`, 2021.
[45] S. Singhal, V. Kavitha, Coalition formation resource sharing games in networks, SIGMETRICS Perform. Eval. Rev. 49 (3) (mar 2022). doi:10.1145/3529113.3529132.
[46] G. Christodoulou, Price of Anarchy, Springer US, Boston, MA, 2008, pp. 665–667. doi:10.1007/978-0-387-30162-4_299.
[47] T. Park, W. Saad, Kolkata paise restaurant game for resource allocation in the internet of things, 2017. doi:10.1109/ACSSC.2017.8335666.
[48] J. Zhang, X. Gao, Z. Yao, Heterogeneous resource management for DAG-based task offloading in satellite networks, IEEE Internet of Things Journal 12 (22) (2025) 48664–48677. doi:10.1109/JIOT.2025.3607133.