This article has been accepted for publication in IEEE Transactions on Network Science and Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2025.3621342

RobGenX: Uncertainty-Aware Inference in eXtreme Computing Power Networks

Rawan F. El-Khatib, *Student Member, IEEE,* Nizar Zorba, *Senior Member, IEEE,* and Hossam S. Hassanein, *Fellow, IEEE*

Abstract*—Computing Power Networks (CPNs) emerged as a promising architectural paradigm that unifies the edge and user-owned eXtreme Edge Devices (XEDs) into an orchestrated compute fabric. By leveraging opportunistic idle resources at the eXtreme Edge (XE), CPNs offer a pathway to democratize generative AI and reduce reliance on cloud monopolies. However, inference delegation within CPNs is hindered by the volatile and user-dependent nature of compute availability of XEDs. In this work, we propose RobGenX, a robust task allocation framework tailored for inference in XE-enabled CPNs under stochastic compute capacity constraints. To the best of our knowledge, RobGenX is the first work to explicitly model the uncertainty of XEDs' compute capacity. RobGenX integrates Recourse Programming (RP) to penalize overassignment and Chance-Constrained Programming (CCP) to enforce probabilistic workload guarantees, respectively. We model the task allocation problem as a two-stage chance-constrained program and obtain an equivalent deterministic reformulation using a scenario-based approach. To overcome the computational burden of the resulting Integer Linear Program (ILP), we develop a decomposition-based solution using the Alternating Direction Method of Multipliers (ADMM), which decouples the relaxed problem into parallelizable subproblems across uncertainty scenarios. Simulation results demonstrate that RobGenX and its lower complexity ADMM-based solution consistently outperform the state-of-the-art task allocation baselines, achieving up to 58% improvement in inference completion rates and 55% in assignment stability.

I. Introduction

A. Background

The emergence of generative Large Language Models (LLMs) has attracted significant attention due to their remarkable capabilities across a wide range of complex and multimodal Artificial Intelligence (AI) tasks [1]. Traditionally hosted in cloud data centers equipped with high-end graphics and/or tensor processing units, these models have inspired a new generation of intelligent services [2]. While inference costs have dropped dramatically over the recent years, running high-performing LLMs at scale remains prohibitively expensive due to their intensive computational requirements [3], [4]. This economic and infrastructural barrier reinforces the monopolistic nature of the generative AI ecosystem, limiting accessibility and slowing innovation [5]. In parallel, visions for 6G networks emphasize low-latency, contextualized AI as a core design goal [6]–[10]. This has given rise to a growing body of research advocating for shifting inference workloads from centralized cloud servers to the eXtreme Edge (XE) of the network [5], [11], [12]. By harnessing the latent compute capacity of everyday user-owned devices, referred to as eXtreme Edge Devices (XEDs), this approach aims to alleviate infrastructure costs, reduce dependence on centralized infrastructure, improve responsiveness, and unlock privacy-preserving, personalized generative AI experiences [5], [11], [12].

To realize the vision of generative AI beyond centralized infrastructures, a flexible and adaptive computing architecture is required to span across edge servers and user-owned end devices. Computing Power Networks (CPNs) address this need by integrating traditional Edge Computing (EC) with eXtreme Edge Computing (XEC) into a unified, orchestrated infrastructure [13]. While EC provisions dedicated compute resources at the network edge (e.g., base stations, gateways), XEC opportunistically leverages idle capacities on personal devices at the network's XE. In scenarios where the XE falls short, the upper EC tier recovers disrupted computations to ensure service continuity. Therefore, CPNs emerge as a foundational enabler of democratized generative AI [14], offering a resilient multi-tiered orchestration framework that blends the cost-efficiency of XEC with the reliability and performance guarantees of EC.

B. Motivation

Recent progress in Small Language Models (SLMs) [15] and the release of open-weight models such as DeepSeek R1 [16] corroborate the growing feasibility of running generative AI on user-owned devices. This is further supported by early commercial deployments of collaborative inference (e.g., EXO Labs[1]) and recent testbeds and evaluations demonstrating practical LLM inference under stringent resource constraints [12], [17], [18]. This positions XE-enabled CPNs as a viable paradigm for democratized generative AI workloads. However, for CPNs to fully unlock their potential, they must be able to orchestrate a highly heterogeneous and volatile pool of XE resources. In particular, achieving inference at the XE requires distributing computation across multiple XEDs. Contrary to centralized cloud servers, XEDs possess limited processing capabilities and exhibit highly dynamic availability patterns due to unique user behavior and fluctuating background workloads [19], [20].

This volatility poses a key challenge to the success of inference task allocation, which is anchored in the accurate

R. F. El-Khatib is with the Department of Electrical and Computer Engineering, Queen's University, Kingston, ON K7L 3N6, Canada (e-mail: rawan.elkhatib@queensu.ca).

N. Zorba is with the College of Engineering, Qatar University, Doha, Qatar (e-mail: nizarz@qu.edu.qa).

H. S. Hassanein is with the School of Computing, Queen's University, Kingston, ON K7L 2N8, Canada (e-mail: hossam@cs.queensu.ca).

[1] https://exolabs.net/

This article has been accepted for publication in IEEE Transactions on Network Science and Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2025.3621342

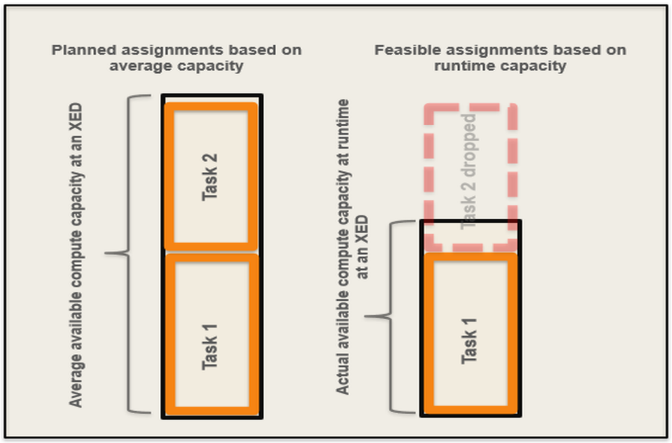

Fig. 1: Illustration of task dropping when deterministic schemes assign tasks based on average capacity, leading to infeasible allocations once runtime capacity is lower than expected. RobGenX mitigates this by incorporating uncertainty into allocation decisions, limiting drop risk, and keeping assignments feasible with high probability.

knowledge of XEDs' available compute capacity. Existing task allocation schemes neglect the volatility in XEDs' compute availability, relying instead on deterministic formulations that assume that average capacity is representative of runtime conditions [21]–[25]. However, this approach degrades performance when the actual available compute at runtime falls below the expected levels, as shown in Fig. 1. Specifically, XEDs may be overassigned beyond their real-time capacity, leading to task dropping, unstable assignment schedules, and lower inference completion rates. This mismatch between anticipated and realized allocations creates a gap between expected and actual gains, undermining the effectiveness of the task allocation.

Besides high computational resource demands, LLM inference is characterized by its stringent quality requirements. Specifically, each task is considered a critical component of the overall computation. The failure of even a single XED to deliver its assigned task jeopardizes the entire inference outcome [25]. CPNs address such disruptions by enabling recovery procedures through fallback to edge servers within their hierarchical orchestration. However, when uncertainty is ignored, a high number of task failures can overwhelm the fallback mechanism, introducing significant delays and disrupting the timely execution of inference, especially in scenarios where edge servers are already overburdened. Simultaneously, participating XEDs experience excessive workload beyond their intended contribution, which increases the likelihood of inference failure and, more importantly, XEDs' disengagement from the system [26]. This in turn shrinks the overall resource pool, undermining the sustainability of collaborative LLM inference at the XE, and, by extension, the orchestration effectiveness of CPNs built to coordinate these distributed resources. As a result, existing schemes that fail to account for the volatile nature of XED compute capacity may suffer significant negative consequences for both service requesters and participating XEDs. These limitations highlight the need for uncertainty-aware task allocation strategies that can maintain performance and engagement despite XE volatility.

To address this gap, we propose RobGenX, a framework for robust generative AI inference at the XE. RobGenX equips CPNs with the stochastic intelligence required to manage inference at the volatile XE layer, enabling informed orchestration decisions that reduce the need for fallback to higher-tier resources such as edge servers. It does so by leveraging stochastic optimization tools to: a) model the compute availability of XEDs as random variables, b) design a task allocation strategy that accounts for the stochastic nature of XED availability and minimizes the penalties incurred from fallback to higher-tier CPN layers, and c) enforce probabilistic constraints that limit workload violations at the XEDs.

C. Contributions

Our contributions can be summarized as follows:

- Uncertainty-aware orchestration in XE-enabled CPNs: We propose RobGenX, a robust task allocation framework designed to support generative AI inference at the XE. To the best of our knowledge, this is the first work to explicitly model the stochastic uncertainty of compute capacity across user-owned XEDs and its detrimental effects to generative AI inference. By treating compute availability as a random variable, RobGenX equips CPN orchestrators with the ability to maximize inference reliability, reduce fallback to higher-tier layers, and sustain engagement of volatile XEDs, even under highly dynamic and unpredictable conditions.
- Stochastic modeling of resource fluctuations: We formulate a two-stage stochastic optimization model that integrates Recourse Programming (RP) and Chance Constrained Programming (CCP) to capture and mitigate the impact of stochastic compute availability. The RP component quantifies the penalty of over-assignment and consequently, fallback to the edge server, while the CCP imposes probabilistic constraints to ensure allocated workloads respect actual idle capacities of XEDs. This enables CPNs to sustain reliable service orchestration across dynamic and uncertain XE layers. To render the model solvable, we utilize the statistical data of XED owners' behaviors to derive a deterministic equivalent reformulation that replaces the original stochastic model, resulting in an Integer Linear Program (ILP) compatible with standard commercial solvers.
- Time-efficient near-optimal solution: To alleviate the computational burden of the ILP, we develop a decomposition-based algorithm that applies the Alternating Direction Method of Multipliers (ADMM) to solve the relaxed deterministic equivalent formulation. We exploit the structure of the problem to reformulate it into a consensus-based structure that decomposes the problem across scenarios. To the best of our knowledge, this is the first work to apply a consensus-based ADMM to solve a two-stage chance-constrained inference task allocation problem in the XE.

TABLE I: Comparison of RobGenX with Existing Work on CPN Hierarchical Collaboration for Generative AI.

Reference	**Target tier**	**Cross-tier collaboration**	**Latency**	**Infrastructure cost**	**Fallback to upper tier?**	**Fallback delay**
Murthy et al. [27]	Cloud	None	High	High	No	N/A
Du et al. [28]	Edge	None	Low	Medium	No	N/A
Li et al. [6]	Edge	None	Low	Medium	No	N/A
Qu et al. [29]	Edge	None	Low	Medium	No	N/A
Jayanath et al. [30]	Edge	None	Low	Medium	No	N/A
Wang et al. [31]	Edge	Cloud–Edge	Medium	Medium	Yes	High
Xu et al. [32]	Edge	Cloud–Edge	Medium	Medium	Yes	High
Tian et al. [33]	Cloud	Cloud–Edge	Medium	Medium	Yes	High
Chen et al. [34]	Edge	Cloud–Edge	Medium	Medium	Yes	High
Shen et al. [5]	XE	None	Low	Low	No	N/A
Zeng et al. [11]	XE	None	Low	Low	No	N/A
Zhang et al. [12]	XE	None	Low	Low	No	N/A
Yang et al. [35]	Cloud	Cloud–XE	Medium	Medium	Yes	High
RobGenX (ours)	**XE**	**Edge-XE**	**Low**	**Low**	**Yes**	**Low**

The proposed approach significantly reduces runtime while preserving near-optimal solution quality.

Extensive performance evaluations show that RobGenX yields significant improvements, achieving an improvement of up to 58% and 55% in the inference completion rate and assignment stability index compared to prominent representatives of task allocation schemes, respectively. The remainder of the paper is organized as follows. Section II presents the background and related work. Section III presents the system model, and Section IV introduces the proposed framework, the stochastic problem model, and the deterministic equivalent form. Section V introduces the ADMM-based solution. Section VI details our performance evaluation experiments. Finally, Section VII discusses practical implementations considerations and Section VIII summarizes our conclusions.

II. Background and Related Work

A. XE-enabled CPNs for Generative AI

XEC has emerged as a transformative computing paradigm, attracting growing attention for its ability to harness the idle computational capacity of user-owned devices at the network's XE [36]. Unlike traditional infrastructure-heavy paradigms, XEC capitalizes on already-deployed devices to enable decentralized, low-cost, and scalable computation. Modern XEDs are equipped with powerful processors and diverse communication interfaces, making them capable of exchanging and processing data in real time. Studies report that battery-powered XEDs typically maintain more than 55% battery availability and leave over two-thirds of their computing power underutilized by local tasks [37]. XEC has demonstrated viability, competing with traditional cloud and EC services with minimal communication and coordination overhead [38]. As such, XEC offers a compelling alternative that alleviates infrastructure costs and disrupts the monopolistic nature of generative AI services [12].

However, despite its promise, XEC alone is not sufficient to guarantee reliable and high-quality generative AI service delivery. The highly dynamic and opportunistic nature of XED availability requires a higher-level coordination mechanism to manage resources effectively. Recently, CPNs have emerged as a key solution to orchestrate compute across cloud, edge, and XE tiers into a unified, collaborative system [7], [13]. In this paradigm, different tiers dynamically cooperate to ensure service continuity, reliability, and cost-efficiency. CPNs facilitate intelligent resource pooling and adaptive scheduling, allowing the system to dynamically offload workloads to the most appropriate tier based on availability and context [13]. For generative AI services, the architectural value of CPNs lies in their ability to mitigate performance bottlenecks and sustain real-time inference quality under volatile XE conditions [33].

Related Work: Several recent works have investigated deploying generative AI at different tiers of the network. Murthy et al. [27] present a solution to enhance generative AI performance in cloud environments through resource scaling, cost containment and model optimization, emphasizing hybrid-cloud strategies for balanced cost, performance, and compliance. While centralized cloud infrastructures provide ample compute and storage, relying exclusively on the cloud presents key limitations, as discussed earlier. To mitigate these issues, several works consider edge-centric generative AI. Du et al. [28] present generative AI as a service and propose a deep reinforcement learning-based service provider selection algorithm to dynamically route users to optimal edge providers. Li et al. [6] consider an AI edge computing framework that integrates computing, sensing, and communication through AI agents that manage both control and user-plane functions. Qu et al. [29] demonstrate the urgent need for deploying LLMs at the network edge, and present an architectural overview of EC for LLMs, outlining its core components and how it supports the deployment of generative AI. Finally, Jayanath et al. [30] consider efficient mapping of computations across multiple cores on an edge server to effectively manage demanding generative AI workloads. These works consider hosting generative AI workloads exclusively on edge servers, bringing computation closer to users. This approach offers latency reductions, improved privacy, and energy efficiency due to localized inference execution. However, deploying and maintaining edge

TABLE II: Comparison of RobGenX with Existing Work on Uncertainty-aware Task Allocation

Reference	Uncertainty	Target tier	Cross-tier collaboration	Solution strategy
Eshragi et al. [39]	Processing cycles	Edge	None	Heuristic solution
Li et al. [40]	Processing cycles	Edge	None	Chance constraints + an online algorithm
Chang et al. [41]	Processing cycles	Edge	None	Chance constraints + a decomposition approach
Ji et al. [42]	Channel conditions and queues sizes	Edge	None	Worst-case optimization + extreme value theory
Nguyen et al. [43]	Service demand and resource size	Edge	Cloud–Edge	Column and constraint generation and the Karush–Kuhn–Tucker conditions
Qin et al. [44]	Channel state information and queuing delays	Vehicular edge	None	Lyapunov optimization + machine learning
Reifert et al. [45]	Channel and compute information	Edge	None	Worst case optimization + deep neural networks
Lai et al. [46]	User arrivals and departures	Cloud–Edge	Cloud–Edge	Lyapunov optimization
Apostolopoulos et al. [47]	User behavior	UAV-assisted edge	None	Prospect Theory
Xia et al. [48]	User mobility, network state	Edge	None	Multi-stage stochastic programming + Lyapunov optimization
Zhang et al. [49]	User mobility	Vehicular edge	None	Markov decision process
Karanika et al. [50]	Task demand and user mobility	Edge	None	Statistical learning and fuzzy logic
RobGenX (ours)	**XED Compute Capacity**	**XE**	**Edge–XE**	**Chance-constraints + recourse programming + ADMM**

infrastructure remains expensive due to the need for specialized hardware, which limits widespread deployment by service providers [51]. Moreover, in high-density scenarios, the finite compute capacity at the edge can quickly become saturated, resulting in service delays or denials during peak usage.

To balance the trade-offs between cloud and edge tiers, collaborative frameworks have emerged. Wang et al. [31] propose that edge-cloud cooperation can offer both computational power and reduced latency for scalable generative AI systems, and highlight challenges and design considerations for deployment at scale using two generative AI applications as examples. Xu et al. [32] conduct a survey on deploying generative AI services in mobile edge networks and explore the implementation, and security and privacy related challenges of cloud–edge–mobile infrastructures. Tian et al. [33] propose a novel bottom-up generative AI architecture that combines large cloud models with lightweight edge models, aiming to address the limitations of centralized deployments and alleviate edge-cloud burdens. Finally, Chen et al. [34] introduce a collaborative cloud–edge framework for personalized generative AI services that distributes tasks between edge LLMs and cloud LLMs, optimizing based on compute capacity and user context. These works focus on cloud–edge cooperation where inference is split across infrastructure layers. These designs partially improve latency and privacy by delegating select tasks to the edge. However, they inherit key drawbacks from both tiers. On one hand, edge servers remain costly and limited in availability, while cloud collaboration constrains performance due to potential network bottlenecks that affect the timeliness of intermediate result exchange.

On the other hand, several recent works advocate for XE-exclusive configurations for generative AI. For instance, Shen et al. [5] argue that the future of generative AI can be democratized through distributed training on massive XEDs. They highlight the limitations of centralized AI scaling in terms of high-quality data and compute monopolies and present XE-enabled generative AI as inevitable. Zeng et al. [11] introduce a collaborative framework, where trusted XEDs pool their idle resources to train generative AI models. They consider scheduling strategies and empirically analyze energy efficiency using different parallelism techniques on real testbeds, establishing a foundation for sustainable and decentralized model training at the edge. Finally, Zhang et al. [12] propose a model-centric framework for deploying generative AI on mobile and edge networks, aiming to overcome the resource limitations of such environments. They survey compact model techniques (quantization, pruning, distillation), evaluate key deployment metrics, and demonstrate feasibility via LLM deployment. These works propose democratized frameworks that capitalize on the ubiquity of XEDs to offload generative AI services from centralized infrastructure. By leveraging idle compute resources on devices that are widely deployed, these approaches eliminate infrastructure costs and bring computation closer to data sources. However, operating exclusively at the XE introduces critical reliability challenges. Specifically, inference execution on XEDs is subject to device volatility and uncertain availability, making it difficult to ensure robust and consistent service quality.

To mitigate such issues, Yang et al. [35] propose a cloud–XE collaborative architecture that balances service-level agreement requirements and cost-efficiency. They design an end-to-end system that involves a greedy scheduling algorithm, a hardware model and a cloud–device compiler framework for deployment. While this improves reliability, it reintroduces several of the drawbacks associated with cloud involvement, such as increased latency, higher costs, and privacy concerns. To the best of our knowledge, our work is the first to consider an XE-Edge CPN architecture that empowers the XE as the primary inference executor, while utilizing nearby edge servers as a recovery layer in failure scenarios. This edge-backed XE configuration uniquely balances cost-efficiency, low latency, and reliability by avoiding costly cloud dependence while still enabling fallback and continuity of service. A summary comparison is provided in Table I.

B. Task Allocation in XEC

Task allocation in XEC has attracted significant attention in the last years [26], [52]–[56]. Conventional task allocation schemes adopt a reactive approach, where allocation decisions are made based on current system states. In particular, reactive schemes use instantaneous compute resource availability as reported by XEDs to optimize task allocation under various objectives. This reactive strategy inherently assumes that the reported resource availability will remain unchanged and that devices operate in a stable environment. However, such assumptions often fail in real-world XEC deployments, as XEDs' resource availability is subject to user-driven dynamicity and unique access behavior [19], [20]. As a result, a task assigned based on an initial resource availability may face disruption, leading to task failure and degraded generative AI performance.

To overcome the limitations of reactive approaches, several studies have shown that it is possible to predict XED availability [20], [57], [58], forming the basis for predictive task allocation schemes in XEC [21]–[25]. While these schemes have demonstrated substantial performance improvements, they rely on point estimates of future availability and assume those predictions to be fully accurate, overlooking the inherent uncertainty present in real-world conditions [19], [20]. Such imperfections are often modeled by utilizing the averages of predicted values, which leads to deterministic formulations that are vulnerable to forecast errors and performance degradation, especially when runtime compute availability falls below the expected values.

In parallel, other research has focused on task allocation under different aspects of uncertainty. Specifically, several works investigate task allocation under uncertain task demand [39]–[41]. Eshragi et al. [39] consider the joint task offloading and resource allocation problem under processing cycle requirement uncertainty. They model the problem as a mixed integer nonlinear program and develop an efficient heuristic algorithm to solve it. Li et al. [40] examine the impact of uncertain processing cycles on computing time and task deadline compliance, formulating a chance-constrained problem aimed at minimizing overall energy consumption. Chang et al. [41] address the problem of minimizing average response time under energy consumption constraints and processing cycle uncertainty for continuous applications, and propose a heuristic algorithm solution. In addition, Nguyen et al. [43] model service demand uncertainty and formulate a stochastic two stage problem to find the optimal location for service placement and the amount of resources to obtain from each location.

On the other hand, Ji et al. [42] examine uncertainties arising from unknown communication channel conditions and network queue sizes. They propose minimizing the worst-case expected energy consumption for time-sensitive applications. Qin et al. [44] analyze channel state information uncertainty in cooperative vehicular offloading systems. They model the joint task splitting, offloading, and computational resource allocation problem and employ Lyapunov optimization to decouple it into tractable subproblems. Similarly, Reifert et al. [45] examine uncertainties in the channel and computing states information, and propose a solution that minimizes the worst-case delay subject to computation and power constraints.

Lai et al. [46] consider uncertainty due to users' arrivals and departures, and formulate the stochastic optimal user to server association with either the cloud or the edge to balance throughput and queuing delay trade-offs. Apostolopoulos et al. [47] consider task allocation in UAV-assisted EC systems. They capture risk-aware behaviour of users when offloading to ground-based and UAV-mounted EC servers, the uncertainties associated with the task allocation via the probability of failure due to the UAV energy constraints when offloading to the UAV, and utilize prospect theory to solve the problem. Xia et al. [48] address uncertainties in user mobility and network states, and propose a stochastic programming algorithm to mitigate the effects of inaccurate predictions. Zhang et al. [49] aim to minimize delay under vehicle mobility uncertainty and utilize Markov decision processes to model and solve the problem. Finally, Karanika et al. [50] propose a geo-referenced task management scheme that utilizes fuzzy logic to represent uncertainties due to users mobility.

The above works examined uncertainties due to task demands [39]–[41], [43], communication resources [42], [44], [45], user behavior [46], [47], and user mobility [48]–[50]. Furthermore, they primarily focus on task allocation to dedicated edge infrastructure with stable and high-capacity compute resources. Consequently, they assume static and predictable availability at the computing edge servers. This assumption is misaligned with the dynamic and user-driven nature of XEC environments. These works do not explicitly consider the uncertainty in compute availability across user-owned XEDs, nor do they examine its impact on the reliability, latency, or sustainability of collaborative inference tasks. In contrast, RobGenX directly tackles this overlooked challenge. To the best of our knowledge, RobGenX is the first work to model XED compute capacity as a random variable and propose a robust allocation framework that incorporates stochastic guarantees for both the service requesters and participating XEDs for generative AI inference tasks. Furthermore, RobGenX extends beyond prior edge-focused solutions by operating within a CPN hierarchy, in which edge and XE tiers collaborate dynamically to maintain service quality. This multi-tier orchestration introduces an additional layer of robustness: when XED-side failures occur, the CPN's edge tier provides a recovery mechanism, preserving inference continuity without resorting to costly or latency-prone cloud intervention. In doing so, RobGenX not only optimizes allocation efficiency and stability but also ensures service sustainability by minimizing device overload and encouraging long-term participation of XEDs in collaborative inference. A summary comparison is provided in Table II.

C. Stochastic Optimization

Stochastic optimization is a powerful modeling paradigm for making decisions in the presence of uncertainty. It embeds random variables directly into the optimization process, allowing the solution to reflect the potential variability of inputs that arise in real-world systems [59]. Unlike deterministic models, which assume all parameters are known with certainty, stochastic models explicitly account for the probabilistic nature of input data. This enables decision-makers to anticipate variability and incorporate risks and consequences directly into the planning process. A central modeling tool within this framework is two-stage programming (also referred to as RP), which partitions decisions into two phases: the initial (first stage) decisions are made before the uncertainty is realized, while recourse (second stage) decisions are taken after the uncertainty has been observed to mitigate its impact. On the other hand, CCP tackles uncertainty in the constraints, where feasibility is enforced with a specified high probability, thus allowing for controlled constraint violations while still capturing the underlying risk.

To solve such stochastic formulations using commercial optimization solvers, the problem must first be transformed into a deterministic equivalent formulation. This transformation replaces the probabilistic components with a finite set of scenarios that approximate the distribution of uncertainty. Such formulations often take the form of ILPs that are notoriously challenging due to the introduction of binary indicator variables, nonconvex feasible regions, and combinatorial complexity [60], [61]. Consequently, such formulations are intractable for direct solution without further structural exploitation. Decomposition algorithms have therefore emerged as essential tools for handling the scale and complexity of these problems [59]. Classical methods such as the L-shaped algorithm have been successfully applied to two-stage linear programs. However, their extension to programs with integer variables is limited. The non-convexity of such problems undermines the validity of dual information. In addition, the L-shaped method requires custom cuts that are not straightforward to obtain.

To address these limitations, we adopt the ADMM, a widely used algorithm for solving optimization problems that exhibit decomposable structure. ADMM decomposes a complex global problem into smaller, more manageable subproblems that can be solved independently. Its robust convergence properties and flexible architecture have enabled its application across a wide range of fields [62]. In the context of stochastic programming, a few studies have applied classical ADMM to solve problems with uncertainty [63]–[65]. However, these efforts utilize the classical ADMM form and do not fully exploit the natural scenario-wise separability inherent in two-stage chance-constrained stochastic programs. In contrast, our work leverages the consensus ADMM framework, which introduces auxiliary consensus variables to decouple scenario-specific subproblems while enforcing agreement through a central coordination step. This enables a distributed architecture in which the global optimization problem is reformulated as a coordination of multiple scenario-local subproblems, each corresponding to a specific realization of uncertainty. To the best of our knowledge, this is the first work to apply consensus ADMM to solve two-stage chance-constrained stochastic task allocation problem for generative AI at the XE.

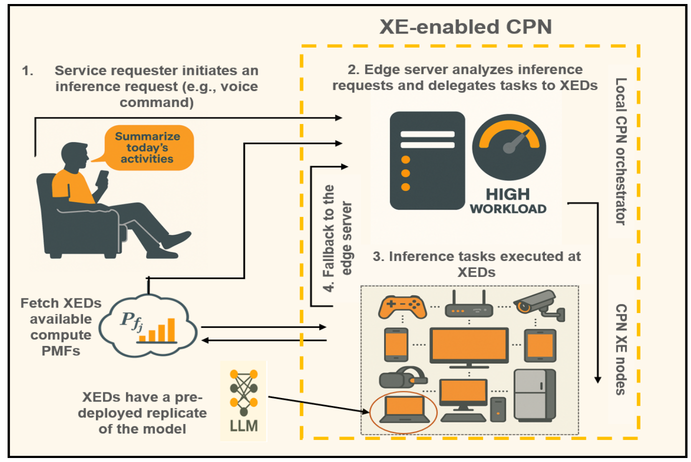

Fig. 2: System model of an XE-enabled CPN where the edge server orchestrates inference by fetching compute profiles from XEDs, delegating tasks to eligible devices with pre-deployed models, and providing fallback execution when needed.

III. System Model

We consider an XE-enabled CPN, where an edge server oversees a large number of user-owned XEDs and acts as the CPN orchestrator. The devices are first assessed via eligibility criteria and filtered according to a fitness function [66], which evaluates their suitability to participate in orchestration by jointly considering compute capacity, residual energy, model availability, and resource reputation. To qualify as a participating worker in the XE-enabled CPN, an XED must (i) have sufficient memory to host a pre-deployed replicate of the model [15], [67] and run the full model independently on a distinct subset of input data [68], (ii) maintain sufficient residual energy and stable connectivity to contribute reliably, and (iii) achieve a threshold fitness value as determined by the orchestrator. Devices that fail to meet these conditions are excluded from orchestration. After this assessment, the orchestrator organizes XEDs into collaborative clusters [68]–[70] by considering factors such as device locality, similarity in resource profiles, and patterns of user requests. This hierarchical structure enables robust orchestration at scale, where task allocation is executed within clusters, while the orchestrator manages requests at the cluster level, ensuring scalability even when thousands of devices are present across the CPN.

A representative application scenario is the smart home, where an edge server embedded within the residential network infrastructure coordinates the execution of generative AI services [68]–[70]. The cluster of smart home XEDs may include smartphones, laptops, tablets, gaming consoles, and smart TVs, each equipped with varying degrees of idle computational capacity. In this setting, the edge server manages generative inference requests initiated by user activity, such as interpreting voice commands to control devices (e.g., turn a lamp on/off), generating personalized content recommendations on smart displays (e.g., reminders about a user's favorite sports team), identifying household devices from room images to enable vision-based control, and issuing context-aware prompts (e.g., reminders to discard expired milk upon fridge access) [69].

Within the CPN architecture, the edge server offloads inference tasks to the cluster of XEDs, leveraging their vast idle compute resources. The benefits of this layered architecture are threefold: a) it maximizes resource utilization at the XE by tapping into the under-utilized computational capacity of the XEDs cluster; b) it alleviates computational pressure on the edge server during high-load periods, reducing the risk of service bottlenecks [71], [72]; and c) it allows to incorporate a fallback mechanism that enables the edge server to execute failed tasks when XEDs become unavailable or underperform, ensuring continuity of inference services despite the volatility of user-owned resources. The system model is illustrated in Figure 2.

Let $\mathcal{I}$ denote the set of inference tasks, indexed by $i \in \mathcal{I}$, and $\mathcal{J}$ the set of XEDs within the smart home cluster, indexed by $j \in \mathcal{J}$. Each task $i \in \mathcal{I}$ is associated with a deadline δ_i by which it must be completed, a computational requirement $c_{i,j}$, an execution delay $D_{i,j}$ and a utility score $g_{i,j}$ that quantifies the allocation gain if task i is assigned to XED j. To facilitate collaborative inference at the XE, XEDs engage in negotiated agreements with the CPN orchestrator, in which they consent to rent portions of their idle computational capacity to execute inference tasks locally on pre-deployed models in exchange for incentives. These incentives may include monetary compensation or access to premium services such as personalized generative AI models. The negotiation process determines both the explicit share of each device's idle capacity allocated to individual tasks, represented by $c_{i,j}$, and the amount of incentives the XED receives as compensation for its resource expenditure in compute, memory, and energy. This ensures that contributions are clearly specified, that resource consumption costs are reflected in the agreement, and that participants receive fair compensation relative to the amount of resources spent [73].

In addition, XEDs commit to sharing statistical profiles and/or usage histories that reflect their recent compute activity over time in a privacy-preserving manner [10]. This enables the CPN orchestrator to maintain a probability mass function (PMF) for each XED $j \in \mathcal{J}$, denoted by $P_{\tilde{f}_j}$, which captures the distribution of the random variable $\tilde{f}_j$ representing the available compute capacity of device j. This modeling aligns with the quantized resource states and discrete system-level reporting granularity inherent to user-owned and mobile platforms. For example, Android devices report CPU utilization and availability through predefined performance states and frequency bins [74]. These profiles can be efficiently estimated offline from historical idle behavior or updated periodically using lightweight profiling mechanisms. We denote the mean of $\tilde{f}_j$ by $f_j = \mathbb{E}[\tilde{f}_j]$, which serves as the expected compute capacity under deterministic approximations.

To formalize task assignment, we define the binary decision variable $x_{i,j} \in \{0,1\}$, where $x_{i,j} = 1$ if task i is assigned to XED j, and $x_{i,j} = 0$ otherwise. The deterministic task allocation optimization problem, based on this formulation, is expressed as an ILP as follows:

$$\max_{x_{i,j}} \quad \sum_i \sum_j g_{i,j} x_{i,j} \tag{1a}$$

$$\text{s.t.} \quad \sum_i c_{i,j} x_{i,j} \le f_j, \quad \forall j \tag{1b}$$

$$\sum_j D_{i,j} x_{i,j} \le \delta_i, \quad \forall i \tag{1c}$$

$$\sum_j x_{i,j} \le 1, \quad \forall i \tag{1d}$$

$$x_{i,j} \in \{0,1\}, \quad \forall i,j \tag{1e}$$

Constraint (1b) ensures that an XED is not overloaded by enforcing that the total assigned workload does not exceed its expected compute availability, as inferred from its historical usage profile. Constraint (1c) guarantees that task deadlines are respected, while (1d) ensures that each task is assigned to at most one XED. As discussed earlier, this deterministic formulation is based on the average of predicted CPU availability values and therefore does not account for the variability introduced by the dynamic and user-driven behavior of XEDs. This model serves as the baseline for the robust stochastic extension presented in the following section.

IV. RobGenX

In this section, we mathematically formulate the problem of RobGenX using stochastic optimization, and then adopt recourse and chance constraint programming to obtain deterministic equivalent forms.

A. Stochastic Formulation

In our formulation, we adopt a two-stage chance-constrained stochastic optimization structure to maximize task allocation gains while simultaneously minimizing the expected losses caused by uncertainty in XED compute availability. Specifically, the first stage captures the task allocation commitments that must be made by the CPN orchestrator prior to the realization of XED compute availability. This is because allocations cannot wait for full knowledge of resource states, as this would introduce unacceptable delays and undermine orchestration [21]. These allocations are governed by probabilistic constraints, ensuring that the assigned workloads remain feasible with high confidence based on statistical capacity profiles. Once the actual compute states of XEDs are revealed, the second stage activates recourse mechanisms that adapt the initial allocations to the realized capacities. These recourse decisions optimize the first-stage outcome by dropping tasks that exceed available resources and redirecting them to fallback. Accordingly, the RP quantifies the expected penalty incurred from these fallback corrections, while the CCP imposes high-confidence probabilistic guarantees to ensure that allocated workloads remain within the actual idle compute capacity of each XED.

To formulate the robust stochastic version of the task allocation, we redefine the decision variable to explicitly denote its role in the first stage of a two-stage stochastic program. Let $x_{i,j}$ represent the first stage binary decision variable indicating whether task i is assigned to XED j before the actual compute availability $\tilde{f}_j$ is realized. The stochastic formulation of the RobGenX problem is given as:

$$\max_{x_{i,j}} \quad \sum_i \sum_j g_{i,j} x_{i,j} - \gamma \mathbb{E}\left[H(x_{i,j}, \tilde{f}_j)\right] \tag{2a}$$

$$\text{s.t.} \quad \mathbb{P}\left(\sum_i c_{i,j} x_{i,j} \le \tilde{f}_j\right) \ge 1 - \beta, \quad \forall j \tag{2b}$$

$$\sum_j D_{i,j} x_{i,j} \le \delta_i, \quad \forall i \tag{2c}$$

$$\sum_j x_{i,j} \le 1, \quad \forall i \tag{2d}$$

$$x_{i,j} \in \{0,1\}, \quad \forall i, j \tag{2e}$$

The objective function consists of two components. The first term captures the anticipated allocation gain incurred by allocating inference tasks to available XEDs. The second term is the RP, which penalizes task dropping arising from overestimating XEDs' compute capacity. The trade-off is governed by the scalar control parameter γ. In parallel, the CCP in Eq. (2b) enforces a probabilistic constraint on capacity violations per XED. The scalar control parameter β governs the allowable probability of workload violation per XED, which can be viewed as a policy parameter selected based on user tolerance to CPU usage and/or service-level agreement constraints. Lower values enforce stricter workload guarantees but may result in more conservative task allocations, whereas higher values allow more aggressive assignments with acceptable risk trade-offs. Note that our formulation can be easily adjusted to feature an XED-specific threshold (i.e., β_j) to reflect heterogeneous XED tolerance to resource violations. For simplicity, we adopt a uniform threshold β over all XEDs, while preserving the ability to guarantee robust service-level assurance under variable conditions.

B. Deterministic Equivalent Formulation

To construct a deterministic equivalent of the robust stochastic formulation presented in Eq. (2), we leverage the statistical compute availability data collected from XEDs. Each XED $j \in \mathcal{J}$ shares CPU usage profiles with the CPN orchestrator, enabling the construction of a PMF $P_{\tilde{f}_j}$ that characterizes the uncertainty in its future compute availability. These PMFs are used to generate a set of scenarios indexed by $n = 1, 2, \dots, N$, where each scenario n represents a joint realization of the idle compute capacities of all XEDs in the system, and occurs with probability $\mathbb{p}_n$. This scenario-based model allows us to reformulate the expectation term in RP into a piecewise linear function. Let $y_{i,j,n} \in \{0,1\}$ denote the second stage fallback decision variable, indicating whether task i, originally assigned to XED j, is dropped in scenario n due to insufficient compute capacity. This triggers a service recovery mechanism within the CPN hierarchy, incurring a fallback penalty $l_{i,j} \ge g_{i,j}$. The deterministic RP reformulation is written as:

$$\min_{y_{i,j,n}} \sum_{n=1}^{N} \mathbb{p}_n \sum_i \sum_j l_{i,j} y_{i,j,n} \tag{3a}$$

$$\text{s.t.} \quad \sum_i (x_{i,j} - y_{i,j,n}) c_{i,j} \le \tilde{f}_j^n, \quad \forall j, \forall n \tag{3b}$$

$$y_{i,j,n} \le x_{i,j}, \quad \forall i, j, n \tag{3c}$$

$$y_{i,j,n} \in \{0,1\}, \quad \forall i, j, n \tag{3d}$$

Here, constraint (3b) ensures that accumulative task demand after task dropping is limited by the realized capacity $\tilde{f}_j^n$. Constraint (3c) ensures that no task is dropped in scenario n unless it was assigned in the first stage.

Similarly, to convert the chance constraint in Eq. (2b) to a deterministic scenario-based reformulation, we adopt the Big-M reformulation strategy proposed in [75]. We introduce scenario-specific binary slack variables $z_{j,n} \in \{0,1\}$ for each XED j and scenario n, and enforce a two-part constraint as shown in Eq. (4).

$$\sum_i c_{i,j} x_{i,j} \le \tilde{f}_j^n + M(1 - z_{j,n}), \quad \forall j, n \tag{4a}$$

$$\sum_n \mathbb{p}_n z_{j,n} \ge 1 - \beta, \quad \forall j \tag{4b}$$

Here, the variable $z_{j,n}$ takes the value of 1 if the resource constraint is satisfied under scenario n, and 0 otherwise, and M is a sufficiently large constant that effectively disables the constraint when $z_{j,n} = 0$, allowing the inequality to be violated in lower probability scenarios. Following the standard guidance in Big-M reformulations [75], [76], we set M to the maximum computational load that can be assigned to a device $M = \sum_{i \in \mathcal{I}} \max_j c_{i,j}$. Constraint (4b) guarantees that the proportion of satisfied scenarios for each XED j meets or exceeds the confidence level $1-\beta$. By combining the scenario-based reformulations of both the RP and CCP components, we arrive at a unified deterministic reformulation expressed as:

$$\max_{x_{i,j},\, y_{i,j,n},\, z_{j,n}} \quad \sum_i \sum_j g_{i,j} x_{i,j} - \gamma \sum_{n=1}^{N} \mathbb{p}_n \sum_i \sum_j l_{i,j} y_{i,j,n} \tag{5a}$$

$$\text{s.t.} \quad \sum_i c_{i,j} x_{i,j} \le \tilde{f}_j^n + M(1 - z_{j,n}), \quad \forall j, n \tag{5b}$$

$$\sum_n \mathbb{p}_n z_{j,n} \ge 1 - \beta, \quad \forall j \tag{5c}$$

$$\sum_j D_{i,j} x_{i,j} \le \delta_i, \quad \forall i \tag{5d}$$

$$\sum_i (x_{i,j} - y_{i,j,n}) c_{i,j} \le \tilde{f}_j^n, \quad \forall j, n \tag{5e}$$

$$y_{i,j,n} \le x_{i,j}, \quad \forall i, j, n \tag{5f}$$

$$\sum_j x_{i,j} \le 1, \quad \forall i \tag{5g}$$

$$x_{i,j},\ y_{i,j,n},\ z_{j,n} \in \{0, 1\}, \quad \forall i, j, n \tag{5h}$$

The deterministic reformulation shown in Eq. (5) is amenable to standard optimization solvers. However, the problem remains an ILP, which is inherently NP-hard [77]. Note that as the number of scenarios increases as a result of fine-grained PMFs or a growing number of XEDs, the size of the search space expands exponentially, making exact solutions computationally expensive. Sampling-based approaches, such as the sample average approximation, are widely adopted to mitigate this issue by reducing the scenario space to a representative subset [75]. Prior work has shown that even for problems with a very large number of scenarios, using a moderate sample size can yield high-quality robust solutions [75]. Nevertheless, even with SAA, the resulting sampled ILP imposes substantial computational overhead due to its combinatorial complexity. To address these limitations, we next develop a distributed optimization solution based on the consensus ADMM algorithm that alleviates the computational burden of the problem while maintaining robust, high quality solutions.

V. Decomposed Consensus ADMM (DC-ADMM) Inference Task Allocation

To relieve the computational burdens posed by the intractable ILP in Eq. (5), it is imperative to develop computationally feasible solutions that offer lower complexity and improved efficiency, which is critical in the context of robust service provisioning for generative AI workloads at the XE. We aim to exploit the structure of the ILP in Eq. (5), which can be seen as two components. The first component comprises the first stage decision variables $x_{i,j}$, which represent task allocation decisions made irrespective of the realization of uncertainty and must remain consistent across all scenarios. The second component comprises the scenario-specific decision variables $y_{i,j,n}$ and $z_{j,n}$, which depend on the realization of the available compute capacity and are subject to local constraints. Given fixed values of the first stage variables, the remaining problem decomposes into a collection of independent subproblems, each associated with a single scenario n. This structure presents an opportunity for distributed optimization, where agents solve a local portion of the problem and coordinate to reach consensus on the first stage decision variables $x_{i,j}$.

In this section, we develop the Decomposed Consensus ADMM (DC-ADMM) task allocation algorithm to enforce cross-scenario decomposition, thereby enabling time-efficient and robust decision-making under uncertainty for generative AI inference at the XE. DC-ADMM relaxes the ILP into a Linear Program (LP) and decomposes it into scenario-specific subproblems solved independently, and iteratively reconciles the shared global variables to ensure consistency across all scenarios. In order to implement the DC-ADMM solution, we must reformulate the optimization problem in Eq. (1) into a form compatible with consensus ADMM. First, we relax the binary decision variables by replacing them with continuous variables in $[0, 1]$, thereby obtaining an LP. This renders the formulation convex. Second, we introduce local agent-specific copies of the first stage decision variables, denoted by $x_{i,j,n}$. We also define each agent's full set of local decision variables as $\mathbf{A}_n \triangleq \left\{x_{i,j,n},\, y_{i,j,n},\, z_{j,n} \;\middle|\; \forall i, \forall j\right\}$. Accordingly, each agent n solves a local task allocation problem based on its scenario's realization $\tilde{f}_j^n$. The contribution of agent n to the global objective captured by the scenario-weighted cost is written as:

$$F_n(\mathbf{A}_n) = -\mathbb{p}_n \sum_i \sum_j \left(g_{i,j} x_{i,j,n} - l_{i,j} y_{i,j,n}\right) \tag{6}$$

Note that the negative sign is introduced to convert the original maximization objective into a minimization problem, which is required for compatibility with the standard ADMM formulation. Additionally, to ensure that each agent respects the feasibility conditions, we define $\Phi(\mathbf{A}_n)$ as an indicator function that encodes the feasibility of the local decision tuple $\mathbf{A}_n$, expressed as:

$$\Phi(\mathbf{A}_n) = \begin{cases} 0, & \text{if } \mathbf{A}_n \in \Gamma \\ \infty, & \text{otherwise} \end{cases} \tag{7}$$

where Γ denotes the set satisfying the constraints (5b), (5d), (5e), (5f), (5g). Such indicator functions are commonly used in ADMM to handle hard constraints indirectly by embedding them into the objective [62]. This allows the algorithm to treat infeasible solutions as having infinite cost, avoiding the need for explicit constraint projection or reformulation.

To coordinate the distributed decisions among agents, we introduce global consensus variables $x_{i,j,0}$ and $z_{j,0}$, which serve as coordination shared variables that link the local subproblems together. Specifically, $x_{i,j,0}$ represents the globally agreed-upon first stage task assignment decision, which must remain scenario-invariant since it is made regardless of the realization of uncertainty. On the other hand, the global variable $z_{j,0}$ is introduced to enforce the chance constraint in Eq. (5c). While each agent independently solves for $z_{j,n}$ to determine whether the capacity constraint for XED j is satisfied under its own scenario realization, the global variable $z_{j,0}$ ensures that the proportion of scenarios that meet the capacity constraint satisfies the probabilistic threshold $1-\beta$ as mandated by the CCP. In this manner, each agent retains full control over its local decision variables $\mathbf{A}_n$ based on its realized compute availability, while the global consensus variables $x_{i,j,0}$ and $z_{j,0}$ capture system-wide constraints that require coordination. The consensus global optimization problem is written as:

$$\min_{x_{i,j,0},\, z_{j,0}} \quad \sum_n F_n(\mathbf{A}_n) + \Phi(\mathbf{A}_n) \tag{8a}$$

$$\text{s.t.} \quad x_{i,j,n} = x_{i,j,0}, \quad \forall i,j,n \tag{8b}$$

$$z_{j,n} = z_{j,0}, \quad \forall j,n \tag{8c}$$

$$x_{i,j,0} \in [0,1], \forall i,j, \quad z_{j,0} \in [0,1], \forall j \tag{8d}$$

The above problem captures the global coordination mechanism of the distributed framework. Its role is to enforce agreement among agents on the shared first stage allocation variables $x_{i,j,0}$ and the scenario-aggregated slack variables $z_{j,0}$, both of which are essential to satisfy the global chance constraints. Specifically, constraint (8b) enforces that all local scenario-specific decisions $x_{i,j,n}$ agree on a common global assignment $x_{i,j,0}$, ensuring consistent task allocation across scenarios, while constraint (8c) aligns the local slack variables $z_{j,n}$ to a shared global value $z_{j,0}$, enabling centralized control over the allowed probability of workload violations. Note that the objective function component $F_n(\mathbf{A}_n)$ represents the original objective distributed across scenarios via the probability-weighted cost terms, and each local variable $x_{i,j,n}$ is a projection of the global first stage variable $x_{i,j,0}$ onto scenario n, while $\Phi(\mathbf{A}_n)$ encodes all feasibility constraints from the original problem, applied independently to each scenario. By adding consensus constraints (8b) and (8c), the coordination problem guarantees consistency across all scenarios, thereby reconstructing the relaxed problem of the original ILP in Eq. (5) using a decomposed and distributed formulation.

To construct the Augmented Lagrangian (AL) of Eq. (8), we introduce dual variables $\lambda^x_{i,j,n}$ and $\lambda^z_{j,n}$, which represent Lagrange multipliers associated with the consensus constraints (8b) and (8c), respectively. In addition, a penalty parameter $\rho > 0$ is used to penalize the disagreement between local and global values in a quadratic manner. The AL is constructed as:

$$\begin{aligned}\mathcal{L}(&\mathbf{A}_1, ..., \mathbf{A}_n, x_{i,j,0}, z_{j,0}, \lambda^x_{i,j,1}, ..., \lambda^x_{i,j,N}, \lambda^z_{j,1}, ..., \lambda^z_{j,N}, \rho) = \\ &\sum_{n=1}^{N} \Big[F_n(\mathbf{A}_n) + \Phi(\mathbf{A}_n) + \sum_{i,j} \lambda^x_{i,j,n}(x_{i,j,n} - x_{i,j,0}) \\ &\quad + \frac{\rho}{2}\sum_{i,j}(x_{i,j,n} - x_{i,j,0})^2 + \sum_j \lambda^z_{j,n}(z_{j,n} - z_{j,0}) \\ &\quad + \frac{\rho}{2}\sum_j (z_{j,n} - z_{j,0})^2 \Big]\end{aligned} \tag{9}$$

We use the ADMM framework to solve Eq. (9) by iteratively updating the local, global and dual variables, as follows. In the kth iteration, each agent n solves its local optimization problem by minimizing the AL with respect to its local variables while keeping the global and dual variables fixed, which is expressed as:

$$\mathbf{A}_n^{(k+1)} = \arg\min_{\mathbf{A}_n \in \Gamma} \mathcal{L}(\mathbf{A}_n, x^{(k)}_{i,j,0}, z^{(k)}_{j,0}, \lambda^{x,(k)}_{i,j,n}, \lambda^{z,(k)}_{j,n}, \rho^{(k)}) \tag{10}$$

Note that the local subproblem balances minimizing the local incurred costs given the compute capacity realization, while penalizing the solution's deviation from the agreed-on consensus values for $x_{i,j,0}$ and $z_{j,0}$. Next, the global first stage task allocation variables are updated. This is achieved by minimizing the AL with respect to the global variables. By deriving Eq. (9) with respect to $x^{(k)}_{i,j,0}$, setting it to zero and rearranging, we obtain:

$$x^{(k+1)}_{i,j,0} = \frac{1}{N}\sum_{n=1}^{N}\left(x^{(k+1)}_{i,j,n} + \frac{1}{\rho^{(k)}}\lambda^{x,(k)}_{i,j,n}\right) \tag{11}$$

Similarly, by deriving Eq. (9) with respect to $z^{(k)}_{j,0}$, setting it to zero and rearranging, we obtain:

$$z^{(k+1)}_{j,0} = \frac{1}{N}\sum_{n=1}^{N}\left(z^{(k+1)}_{j,n} + \frac{1}{\rho^{(k)}}\lambda^{z,(k)}_{j,n}\right) \tag{12}$$

Finally, the dual variables update can be written as:

$$\lambda^{x,(k+1)}_{i,j,n} = \lambda^{x,(k)}_{i,j,n} + \rho^{(k)}(x^{(k+1)}_{i,j,n} - x^{(k+1)}_{i,j,0}) \tag{13a}$$

$$\lambda^{z,(k+1)}_{j,n} = \lambda^{z,(k)}_{j,n} + \rho^{(k)}(z^{(k+1)}_{j,n} - z^{(k+1)}_{j,0}) \tag{13b}$$

Now that the primal and dual variables are obtained, we check convergence at iteration k as follows. First, we obtain the primal and dual residuals as [62]:

$$r_p^{x,(k+1)} = \sum_{n=1}^{N} \|x^{(k+1)}_{i,j,n} - x^{(k+1)}_{i,j,0}\|_2^2 \tag{14a}$$

$$r_p^{z,(k+1)} = \sum_{n=1}^{N} \|z^{(k+1)}_{j,n} - z^{(k+1)}_{j,0}\|_2^2 \tag{14b}$$

$$r_d^{x,(k+1)} = N\rho^2\|x^{(k+1)}_{i,j,0} - x^{(k)}_{i,j,0}\|_2^2 \tag{14c}$$

$$r_d^{z,(k+1)} = N\rho^2\|z^{(k+1)}_{j,0} - z^{(k)}_{j,0}\|_2^2 \tag{14d}$$

In particular, the primal residuals in Eq. (14a) and (14b) ensure consistency between each agent's local decision and the global consensus, while the dual residuals in Eq. (14c) and (14d) measure the stability of the global variables across iterations. Next, the quadratic penalty variable ρ is updated as [78]:

$$\rho^{(k+1)} = \begin{cases} \tau\rho^{(k)} & \text{if } \|r_p^{x,(k+1)}\|_2 > \mu\|r_d^{x,(k+1)}\|_2 \\ \rho^{(k)}/\tau & \text{if } \|r_d^{x,(k+1)}\|_2 > \mu\|r_p^{x,(k+1)}\|_2 \\ \rho^{(k)} & \text{otherwise} \end{cases} \tag{15}$$

where τ and μ are constants that control the update process. This accelerates convergence and maintains numerical stability.

Specifically, if the consensus gap as indicated by the primal residual is significantly larger than the dual residual, the penalty is increased to force tighter agreement. Conversely, if the dual residual dominates, the penalty is relaxed to allow more exploration. This ensures a balanced and adaptive trajectory towards convergence, as recommended in the literature [62]. To evaluate convergence, we calculate the stopping thresholds as:

$$\epsilon^x_{\text{pri}} = \epsilon^{\text{rel}} \max\left\{\max_n \left\|x^{(k+1)}_{i,j,n}\right\|_2,\ \left\|x^{(k+1)}_{i,j,0}\right\|_2\right\} + \sqrt{N}\,\epsilon^{\text{abs}} \tag{16a}$$

$$\epsilon^z_{\text{pri}} = \epsilon^{\text{rel}} \max\left\{\max_n \left\|z^{(k+1)}_{j,n}\right\|_2,\ \left\|z^{(k+1)}_{j,0}\right\|_2\right\} + \sqrt{N(1-\beta)}\,\epsilon^{\text{abs}} \tag{16b}$$

$$\epsilon^x_{\text{dual}} = \epsilon^{\text{rel}} \left\|\lambda^{x,(k+1)}_{i,j,n}\right\|_2 + \sqrt{N}\,\epsilon^{\text{abs}} \tag{16c}$$

$$\epsilon^z_{\text{dual}} = \epsilon^{\text{rel}} \left\|\lambda^{z,(k+1)}_{j,n}\right\|_2 + \sqrt{N(1-\beta)}\,\epsilon^{\text{abs}} \tag{16d}$$

where ϵ^{abs} and ϵ^{rel} are control parameters [62]. The choice of dynamic stopping thresholds allows the convergence thresholds to adapt according to the variable magnitudes and the desired accuracy mandated by ϵ^{abs} and ϵ^{rel}. Note that in Eq. (16b) and (16d), the stopping thresholds ϵ^z_{pri} and ϵ^z_{dual} accommodate some slack variability across agents to avoid overly pessimistic task allocation decisions by scaling to $(1-\beta)$. This ensures that the probabilistic guarantee in the original problem is met at the $1-\beta$ threshold, without enforcing excessive conservatism by adding more scenarios than necessary. The DC-ADMM task allocation algorithm is shown in Algorithm 1. Consensus ADMM is guaranteed to converge under convexity for any choice of positive ρ [78]. In our case, this guarantee applies to the LP-relaxed formulation, where binary decision variables are replaced by continuous variables in $[0,1]$, yielding a convex feasible set. The DC-ADMM algorithm therefore inherits this convergence property. In practice, after convergence of the relaxed problem, integrality of the decision variables is restored through rounding and feasible repair procedures, ensuring that the final task allocation respects binary requirements.

The time complexity of the DC-ADMM task allocation algorithm is analyzed as follows. In each iteration, each local subproblem is solved independently in parallel across the N agents, as described in lines 3–6. Then, the complexity per agent is $\mathcal{O}(I^3J^3)$. The global coordination step, which involves computing the average of local variables in line 7, incurs a cost of $\mathcal{O}(IJN + JN)$. The dual variable updates in line 8 also require $\mathcal{O}(IJN + JN)$ operations. Additionally, evaluating the primal and dual residuals in line 9 adds another $\mathcal{O}(IJN+JN)$, and the update of the penalty parameter ρ in line 10 involves only constant-time operations giving $\mathcal{O}(1)$. Therefore, the total per-iteration complexity of the DC-ADMM algorithm is dominated by the local convex subproblems, yielding an overall iteration cost of $\mathcal{O}(I^3J^3)$. In contrast, solving the original ILP formulation in Eq. (5) centrally incurs exponential time complexity of $\mathcal{O}(2^{IJ(N+1)+JN})$ [77], highlighting its intractability. The proposed DC-ADMM method significantly improves the computational efficiency by decomposing computation and enabling parallel execution across agents.

Algorithm 1: The DC-ADMM Inference Task Allocation Algorithm

1: **Initialize:** Variables $x^{(0)}_{i,j,n}$, $z^{(0)}_{j,n}$, $\lambda^{x,(0)}_{i,j,n}$, $\lambda^{z,(0)}_{j,n}$, set iteration counter $k = 0$
2: **while** $k < k_{\text{max}}$ or until convergence **do**
3: **Step 1:** Update local variables
4: **for** $n = 1$ to N **do**
5: Solve agent subproblem using Eq. (10)
6: **end for**
7: **Step 2:** Update global variables using Eq. (11) and (12)
8: **Step 3:** Update dual variables using Eq. (13a) and (13b)
9: **Step 4.1:** Compute primal and dual residuals using Eq. (14a), (14b), (14c), (14d)
10: **Step 4.2:** Update penalty parameter using Eq. (15)
11: **Step 4.3:** Compute stopping thresholds using Eq. (16a), (16b), (16c), (16d)
12: **Step 5:** Check Convergence
13: **if** $r^{x,(k+1)}_p \le \epsilon^x_{\text{pri}}$ and $r^{z,(k+1)}_p \le \epsilon^z_{\text{pri}}$ and $r^{x,(k+1)}_d \le \epsilon^x_{\text{dual}}$ and $r^{z,(k+1)}_d \le \epsilon^z_{\text{dual}}$ **then**
14: Convergence Achieved. Terminate.
15: **end if**
16: Increment iteration counter: $k \leftarrow k+1$
17: **end while**

VI. Performance Evaluation

In this section, we evaluate the performance of the ILP and the DC-ADMM solutions in comparison to three representatives of the state-of-the-art task allocation strategies. The first scheme, referred to as the Deterministic Task Allocation Scheme (DTAS), is a representative of a prominent class of prediction-based approaches that utilize forecasts of compute capacities [21]–[24]. DTAS adopts a deterministic formulation where the compute capacity of each XED is represented by the expected value of its PMF. For fairness of comparison, DTAS solves the optimization problem presented in Eq. (1) and serves as a baseline that embeds the average of XEDs computational availability. The second scheme is referred to as the Hypothetical Knowledge Task Allocation Scheme (HKTAS), which models a hypothetical case where the orchestrator has perfect foresight of each XED's actual available compute capacity. Unlike DTAS, which relies on the expected values, HKTAS solves the task allocation problem using the true (sampled) values of each XED's available capacity. Thus, HKTAS represents an upper bound on the achievable performance gains. Lastly, the Worst-case Scenario Task Allocation Scheme (WTAS) resembles a widely-adopted conservative approach that prioritizes safeguarding task execution by optimizing for the worst-case realization of uncertainty [42], [45]. To ensure a fair comparison, WTAS solves the same optimization problem

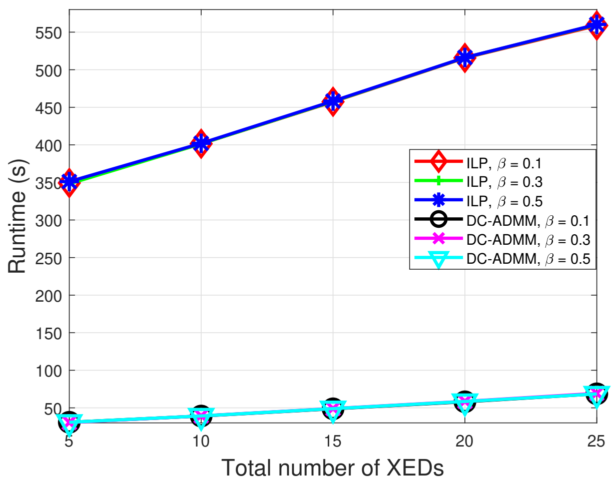

Fig. 3: Runtime comparison of RobGenX using DC-ADMM against the ILP baseline under varying numbers of XEDs and β values, showing the scalability advantage of the distributed approach.

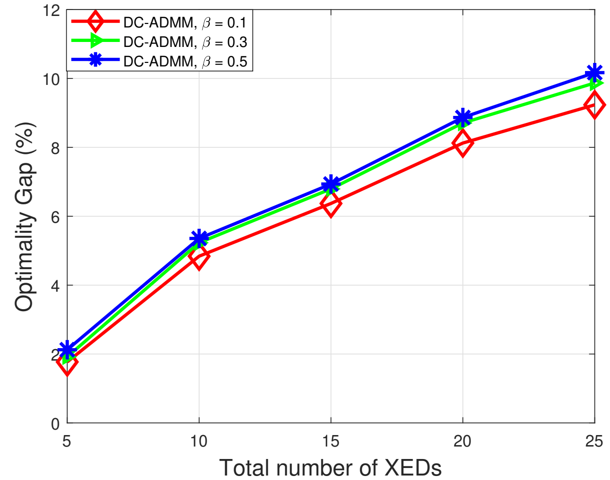

Fig. 4: Optimality gap of DC-ADMM across different numbers of XEDs and β values, demonstrating near-optimal performance with increasing network scale.

as in Eq. (1) with the available capacity per XED substituted by the minimum value in its PMF.

To substantiate the viability of the DC-ADMM solution, we evaluate both its runtime speedup and optimality gap compared to the extensive ILP solution. Additionally, to assess the effectiveness of the uncertainty-aware task allocation framework, we employ the following performance metrics. First, the Inference Completion Rate (ICR) measures the fraction of inference tasks that are assigned and successfully executed by XEDs, relative to the total number of inference tasks in the system. Second, the Assignment Yield Gap (AYG) quantifies the percentage difference between anticipated and achieved inference allocation gains, capturing the portion of intended offloading benefit lost due to uncertainty in XED compute availability. In addition, the Assignment Stability Index (ASI) represents the proportion of task assignments that remain valid at runtime despite the uncertainty of XED compute availability reflecting the robustness of the allocation. Finally, the Fallback Response Time (FRT) captures the accumulative processing and queuing delays incurred at the edge server when inference tasks fail to execute at the XE due to insufficient compute capacity and are subsequently redirected for completion.

A. Simulation Setup

The ILP, DC-ADMM, DTAS, HKTAS and WTAS are all implemented using MATLAB. Simulation experiments are carried out in an area of 200m × 200m, within which XEDs are uniformly distributed. The number of XEDs within the cluster varies in $\{5, 10, ..., 25\}$, each representing a user-owned device with stochastic compute capacity [68], [69], [79]. Each XED j is associated with a PMF with a mean of 3 GHz and standard deviation in $[0.5, 0.9]$ GHz. XEDs' arrival times follow a Poisson process with $\lambda = 0.1$. The total number of inference tasks is set to 75, where each inference task is characterized by an input data size uniformly distributed in $[16, 64] \times 10^3$ bits, and a computational requirement $c_{i,j}$ is uniformly sampled in $[0.5, 1.5]$ GHz.

In the DC-ADMM solution, the maximum number of iterations is set to $k_{\max} = 50$. The algorithm initializes the global and local primal variables $(x^{(0)}_{i,j,n},\ z^{(0)}_{j,n}, x^{(0)}_{i,j,0},\ z^{(0))}_{j,0}$ and the dual variables $(\lambda^{x,(0)}_{i,j,n},\ \lambda^{z,(0)}_{j,n})$ to zero. The penalty parameter ρ is initialized to 1 and is adaptively tuned at each iteration based on the relative magnitude of the primal and dual residuals, using the dynamic adjustment scheme governed by parameters $\mu = 10$ and $\tau = 2$ [62]. The residual thresholds for convergence ϵ^{rel} and ϵ^{abs} are set to 10^{-2}.

B. Simulation Results and Discussion

We evaluate the performance of DC-ADMM against the extensive ILP formulation in terms of runtime speedup and optimality gap. All experiments are repeated 50 times for each instance, and the simulation results are presented with a 90% confidence level.

Figure 3 depicts the average runtime when the number of inference tasks is 75 while the number of XEDs varies. The runtime of the ILP is consistently higher than that of the DC-ADMM solution, and it increases noticeably as more XEDs are introduced. On the other hand, DC-ADMM achieves substantial runtime reductions even as the number of XEDs grows. The linear upward trend of the DC-ADMM runtimes confirms the effectiveness of the scenario-wise decomposition and parallelization approach to alleviate the computational burden. This ensures practical real-time orchestration in CPNs where rapid decision-making across volatile XEDs is essential to ensure continuity of generative AI inference. Notably, variations in the values of the CCP control parameter β have negligible impact for both the ILP and DC-ADMM solutions. This can be attributed to the relative roles that the decision variables $x_{i,j,n}$ and $z_{j,n}$ play in the problem structure. Recall from Eq. (5) that the variable $x_{i,j,n}$ governs the inference task assignment

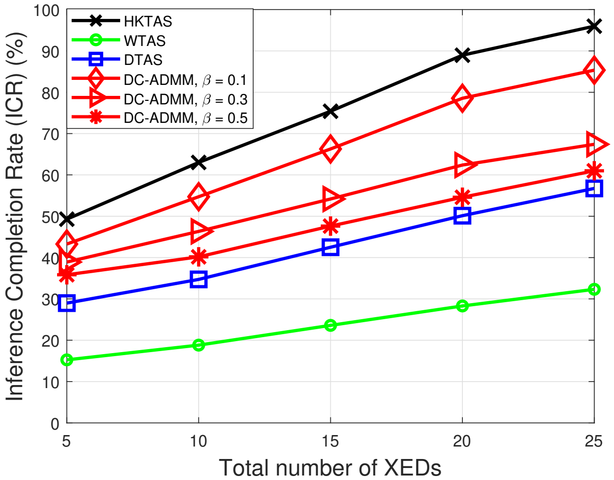

Fig. 5: Inference Completion Rate (ICR) comparison of DC-ADMM with HKTAS, DTAS, and WTAS under varying numbers of XEDs, showing that DC-ADMM consistently achieves higher task completion as network scale increases.

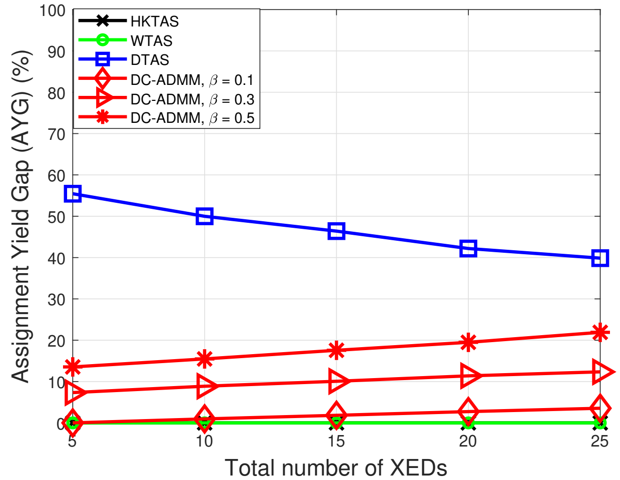

Fig. 6: Assignment Yield Gap (AYG) comparison of DC-ADMM with HKTAS, DTAS, and WTAS across different numbers of XEDs, indicating that DC-ADMM maintains low yield gaps relative to heuristic baselines.

and is directly involved in the objective function and the core constraints, whereas the decision variable $z_{j,n}$ governs the proportion of scenarios in which workload violations are allowed and primarily serves as a near-consensus indicator across scenarios. For the ILP, variations in β impact what constitutes a feasible solution, but do not alter the structure or dimensionality of the problem. In the DC-ADMM experiments, it was observed that the $z_{j,n}$ decision variables typically reach $1-\beta$ consensus earlier than the $x_{i,j,n}$ variables, while convergence is contingent on both variables achieving the target consensus. Therefore, the runtime for DC-ADMM hinges on the consensus of the $x_{i,j,n}$ decision variable, which is unaffected by variations in β.

Figure 4 presents the average optimality gap of the DC-ADMM solution relative to the extensive ILP formulation as the number of XEDs increases. The optimality gap grows gradually with the problem size, since larger numbers of XEDs introduce a larger solution space and a more complex consensus landscape. However, the gap remains consistently below 10% across all configurations, thereby highlighting the ability of DC-ADMM to achieve high-quality approximations with significantly lower computational cost. It is observed that the DC-ADMM optimality gap exhibits a mild sensitivity to the CCP control parameter β. Lower values of β (e.g., $\beta = 0.1$) result in slightly smaller optimality gaps compared to higher values (e.g., $\beta = 0.5$). This is due to the stricter enforcement of scenario-wide consensus over workload feasibility at smaller β, which reduces the flexibility of the solution space and pushes the algorithm towards more conservative and globally aligned decisions.

On the other hand, larger values of β relax the probabilistic constraint (5c) to be more permissive and allow greater variability in the feasible workload allocations, resulting in greater deviation from the optimal solution. In general, the observed differences are negligible, indicating that DC-ADMM maintains a reliable solution quality across variable stochastic tolerance settings, allowing the CPN orchestrator to balance task allocation optimality and runtimes, thereby supporting sustained, cost-effective inference at the XE without over-relying on fallback mechanisms. Given that DC-ADMM consistently achieves high-quality solutions with a small optimality gap and substantially lower runtime compared to the extensive ILP formulation, we adopt the DC-ADMM as the primary solution method in the subsequent experiments to evaluate the behavior of uncertainty-aware task allocation schemes.

Figure 5 illustrates the ICR achieved by DC-ADMM and the three baselines as the total number of XEDs increases. The ICR shows a consistent upward trend across all schemes as the number of XEDs increases. This is due to the expanded resource pool and as a result, the increased scheduling flexibility. HKTAS consistently yields the highest completion rates, aligned with its role as an upper bound due to its perfect foresight into the future computational availability of XEDs. The performance of the DC-ADMM solution closely follows, with ICR values increasing steadily across all values of β. Lower values of β (e.g., $\beta = 0.1$) consistently achieve higher completion rates compared to more relaxed settings (e.g., $\beta = 0.5$). This aligns with the stricter enforcement of workload feasibility under smaller β, which induces more conservative task assignments and imposes tighter chance constraints on the probabilistic workload feasibility condition, effectively reducing the likelihood of assigning workloads that exceed the actual observed available capacity of XEDs. As a result, fewer instances of over-assignment are allowed, and subsequently fewer task droppings due to capacity violations are observed at runtime. This directly improves the success rates of inference tasks. Conversely, higher values of β relax the workload constraint, tolerating more instances of over-assignment and making the task assignment more prone to

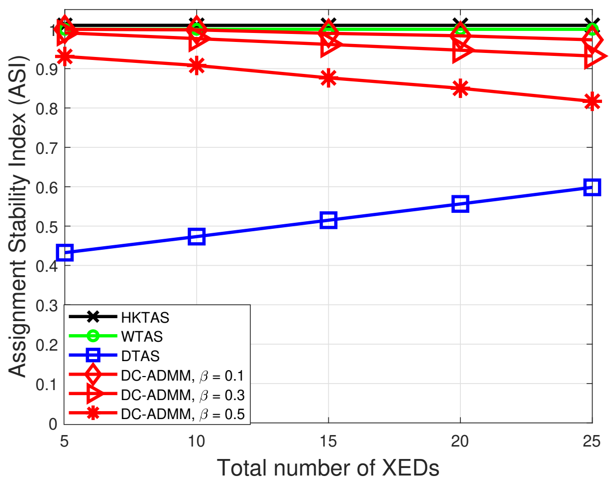

Fig. 7: Assignment Stability Index (ASI) comparison of DC-ADMM with HKTAS, DTAS, and WTAS under varying numbers of XEDs, showing that DC-ADMM achieves high stability across scales and outperforms DTAS.

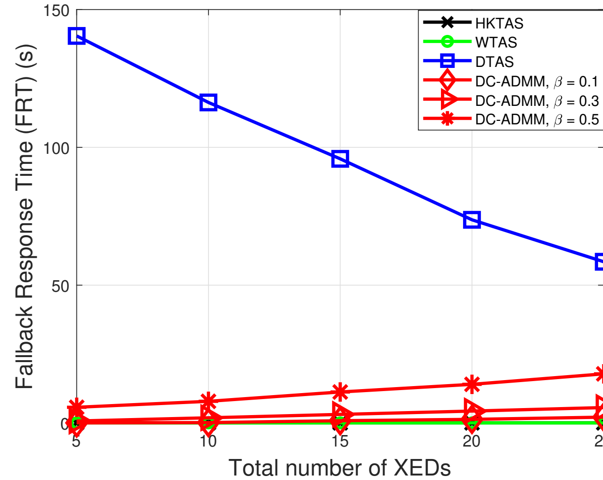

Fig. 8: Fallback Response Time (FRT) comparison of DC-ADMM with HKTAS, DTAS, and WTAS across different numbers of XEDs, demonstrating that DC-ADMM maintains low response times, unlike DTAS which incurs significantly higher delays.

failure under uncertain resource availability.

The DTAS baseline exhibits moderate performance but remains consistently below all DC-ADMM configurations. This highlights the limitations of average-based deterministic approximations under dynamic resource variability. In particular, DTAS fails to safeguard against scenarios where the actual compute capacity of XEDs falls below the expected value used in its formulation. Since DTAS optimizes based on the average capacity derived from the PMF of each XED, it is susceptible to over-assignment when realizations are unfavorable. This increases dropped inference tasks and reduced completion rates. Interestingly, its performance becomes comparable to that of DC-ADMM under the most relaxed stochastic tolerance setting ($\beta = 0.5$), where the chance constraints allow greater flexibility in workload assignment. This suggests that when probabilistic guarantees are loosened, the behavior of DC-ADMM begins to resemble that of DTAS in terms of permissiveness, resulting in similar assignment patterns and failure rates. However, unlike DTAS, DC-ADMM maintains formal control over workload violations through the tunable parameter β, offering an explicit adaptive mechanism for uncertainty management. Finally, the WTAS scheme exhibits the lowest ICR across all XED counts due to its overly conservative treatment of capacity uncertainty, which severely limits feasible task assignment. These results demonstrate that the DC-ADMM approach offers a strong trade-off between robustness and utilization, outperforming existing schemes while remaining close to the theoretical upper bound.

Figure 6 presents the AYG as the number of XEDs varies. Recall that the AYG quantifies the discrepancy between the anticipated allocation gains, reflecting the extent to which a given task allocation strategy is susceptible to over-estimation and task dropping due to resource uncertainty or capacity mismatch. As expected, HKTAS achieves an AYG of zero across all values of XEDs. This is because it operates with perfect foresight and produces zero-error task allocations. WTAS also exhibits the same behavior, due to its conservative approach that relies on the worst-case value in the PMF to prevent overcommitment. In contrast, DTAS suffers from the highest AYG across all schemes, starting above 55% and gradually decreasing as more XEDs are introduced in the resource pool. This is due to to load balancing that inherently reduces the likelihood that an XED will be overburdened relative to its average availability.

All DC-ADMM configurations maintain substantially lower AYG than DTAS, confirming the benefit of stochastic modeling of uncertainty in compute capacity. Note that the AYG increases as β becomes more relaxed, increasing from just below 5% when $\beta = 0.1$, to exceeding 20% when $\beta = 0.5$. This is due to the CCP permitting more optimistic inference task allocations that are prone to dropping at runtime. Finally, while DTAS benefits from an increased number of XEDs, DC-ADMM experiences a degradation in its performance and exhibits an upward trend in AYG as more XEDs are introduced. This is due to the inherent exposure to higher levels of uncertainties, which amplify the variability in feasible task assignments. Furthermore, note that this trend is more pronounced for higher values of β. Recall that larger β values permit greater flexibility in violating workload constraints across scenarios, which can lead to over-optimistic task assignments that deviate substantially from the optimal solution. This makes the task allocation more vulnerable to capacity over-estimation. This underscores the importance of enforcing stricter consensus when operating under high uncertainty, where a lower β can help contain the effects of increased uncertainty and maintain robust alignment across scenarios, allowing the CPN orchestrator to deliver consistent, high-quality inference across fluctuating resource conditions at the XE layer.

Figure 7 illustrates the ASI across varying numbers of XEDs.

Recall that the ASI captures the fraction of task assignments that remain valid at runtime despite fluctuations in compute availability, serving as an indicator of solution robustness. As expected, both HKTAS and WTAS exhibit perfect stability across all configurations. HKTAS performs with perfect foresight, ensuring that every task assignment is feasible at runtime. In contrast, WTAS guarantees stability by design, as it intentionally under-utilizes XEDs capacities using worst-case realizations, thereby eliminating any risks of over-assignment. The tradeoff is shown with comparison of Figure 5, highlighting a significant loss in ICR. DC-ADMM achieves comparable levels of ASI, where β=0.1 consistently approaches the stability levels of HKTAS and WTAS. Stable task assignments reinforce CPN resilience by reducing the need for fallback and ensuring prolonged participation of XEDs, which serves the long-term sustainability of collaborative inference. In contrast, DC-ADMM with β=0.5 shows a notable decline in ASI as the number of XEDs increases, highlighting the vulnerability of the relaxed CCP to task droppings at runtime. DTAS demonstrates an upward trend in ASI as more XEDs are introduced. This improvement, as discussed earlier, reflects the effect of load distribution across a larger pool, thereby mitigating the impact of over-estimations.

Figure 8 reports the FRT across varying number of XEDs. The FRT accounts for both the queuing and processing delays failed inference tasks experience at the edge server, which is typically overburdened and not optimized for direct inference task execution under high-load conditions. This fallback strategy is key to achieving CPN reliability, by ensuring inference task execution even if it fails at the XE. These failures stem from over-assignment instances where the assigned XEDs lack sufficient compute capacity to process their inference tasks, often resulting from optimistic task allocations that fail to account for resource uncertainty. As anticipated, the DTAS scheme exhibits the highest FRT, especially when the number of XEDs is low. This is directly linked to the increased task dropping rates observed under DTAS as shown in Figure 7, which are attributed its reliance on average capacity estimates and lack of probabilistic safeguards. As the number of XEDs increases, the FRT under DTAS declines due to load balancing. Nonetheless, it remains significantly higher than all other schemes.

In contrast, HKTAS and WTAS maintain zero FRT values across all configurations, as their respective approaches completely eliminate runtime task dropping. The proposed DC-ADMM solution achieves substantial reductions in FRT, where lower values of β (e.g., β=0.1) consistently result in lower FRT, reflecting the benefit of conservative allocations in avoiding overloading and minimizing fallback to the edge server. Notably, DC-ADMM with β=0.5 demonstrates a gradual increase in FRT as the number of XEDs grows. This indicates that while a larger pool of XEDs provides more assignment flexibility, insufficiently constrained task allocations under higher β values increase the likelihood of inference failures, triggering fallback to the edge server. This shifts computational burden back to the edge layer of the CPN, thereby diminishing the responsiveness gains that XE-enabled CPNs are designed to deliver.

VII. Implementation Considerations and System Integration in CPNs

In our system, the participating cluster of XEDs share their PMFs representing idle compute capacity with the CPN orchestrator. To preserve security and protect the privacy of individual users and their devices, blockchain-assisted cooperation and coordination mechanisms can be adopted, as discussed in [10]. However, in real deployments, statistical profiles may be imperfect and/or incomplete due to reporting inaccuracies or privacy-preserving restrictions. RobGenX addresses this challenge by relying on probabilistic guarantees that tolerate uncertainty in reported capacities, and by embedding recourse penalties to account for mismatches. Privacy-preserving methods such as the one discussed in [10] reduce data disclosure while still supporting robust orchestration. Furthermore, incentive mechanisms [73] can be designed to encourage accurate participation, ensuring that robustness is maintained even under privacy constraints.

Given XEDs' PMFs, RobGenX operates at the cluster level as follows. Consider a scenario where a user initiates a smart home inference task through a home automation app by submitting a request, such as: "Optimize my appliance schedule to minimize tomorrow's electricity bill." The CPN orchestrator analyzes the request, and triggers a multi-modal generative AI pipeline involving data sources such as smart meter readings, dynamic electricity pricing forecasts, appliance usage logs, and occupancy patterns. The CPN orchestrator decomposes the request into four inference tasks: a) time-series analysis of electricity pricing data, b) load estimation for home appliances using recent usage patterns, c) prediction of occupancy windows based on historical behavior, and d) generation of an optimized appliance usage schedule using a generative model.

Next, the CPN orchestrator formulates and solves the RobGenX task allocation problem in Eq. (8) using the distributed DC-ADMM approach depicted in Algorithm 1, with the CCP conservativeness parameter set to $\beta = 0.1$ to reduce the likelihood of task overassignment. As demonstrated in our simulation results (Figure 3 and Figure 4), DC-ADMM obtains a near-optimal solution with significant runtime reductions compared to solving the full ILP. After solving the task allocation problem, the assigned XEDs execute the inference tasks. Because of the probabilistic guarantees enforced by $\beta = 0.1$, fallbacks to the edge server are rare. Simulation results confirm that this configuration maintains a high degree of assignment stability, as depicted in Figure 7. Moreover, Figure 8 shows that the fallback response delay remains negligible even when failures occur, reinforcing the value of RobGenX in supporting robust CPN orchestration across the volatile XE.

In some scenarios, an XED within a cluster may possess the required input data (e.g., smart meter logs stored on a home energy monitor or occupancy data logged on a personal smartphone) but lack the computational capacity to execute the assigned task. To preserve service continuity in such cases, RobGenX can employ secure data offloading and task migration techniques (e.g., [80]), allowing the input data to be transmitted from a resource-constrained XED to a neighboring device with sufficient computational capacity.

To facilitate secure and efficient communication between the CPN orchestrator and participating cluster of XEDs, the Constrained Application Protocol (CoAP)[2] offers a practical solution. CoAP is standardized by the Internet Engineering Task Force (IETF), and is specifically designed for constrained environments such as smart homes. Its support for Datagram Transport Layer Security (DTLS) ensures end-to-end data confidentiality and integrity with minimal communication overhead. CoAP's asynchronous messaging and multicast capabilities align with the decentralized orchestration logic of RobGenX, enabling efficient broadcasting of task assignments and fallback requests to multiple XEDs. Furthermore, CoAP's built-in resource discovery mechanisms allow the orchestrator to dynamically identify and interact with available XEDs in real time.

Specifically, CoAP enables the XEDs to report their compute capacity by sending PMFs to the orchestrator using POST /compute_capacity requests. To account for device sustainability, XEDs can also periodically report their residual energy through POST /energy_status requests. This allows the orchestrator to incorporate energy availability into orchestration decisions, preventing overburdening of low-battery devices and ensuring fair task distribution. Once the task allocation decision is made, the edge server notifies each XED of its assigned job via POST /assign_task, with the payload detailing the required inference operation. In cases where one XED holds the necessary data but lacks sufficient compute, secure inter-device communication is achieved using PUT /data_transfer requests, and is safeguarded by OSCORE's[3] encryption to preserve confidentiality and integrity. CoAP's message types CON and ACK are used to confirm and acknowledge reliable delivery of critical messages, such as task assignments and data transfers. Finally, the CPN orchestrator can periodically query device capabilities using GET /capabilities to maintain an up-to-date view of the available compute landscape.

VIII. Conclusions and Future Work

In this paper, we presented RobGenX, a novel stochastic task allocation framework for generative AI tasks within XE-enabled CPNs. RobGenX explicitly models the uncertainty in compute capacity across user-owned XEDs, a challenge that has been largely overlooked in prior work. To capture the impact of such uncertainty on inference reliability, RobGenX formulates a stochastic optimization problem that integrates RP and CCP. This formulation allows us to strike a tunable balance between maximizing inference task gains and minimizing the risk of workload violations across participating XEDs. RobGenX enhances the network's ability to coordinate and efficiently harness heterogeneous compute resources at the XE.

[2]https://datatracker.ietf.org/doc/html/rfc7252
[3]https://datatracker.ietf.org/doc/rfc8613/

We proposed a distributed optimization algorithm, referred to as DC-ADMM, which builds on ADMM and decomposes the stochastic problem across scenarios in a consensus-based manner. We validated the effectiveness of the DC-ADMM solution by comparing it against the RobGenX extensive formulation, showing that it achieves high-quality solutions with a small optimality gap and substantial reductions in runtime. In addition, we conducted extensive performance evaluations of DC-ADMM in comparison to state-of-the-art baselines, including prediction-based, worst-case scenario, and hypothetical task allocation schemes. Simulation results demonstrated that DC-ADMM consistently outperforms the deterministic and worst-case schemes in terms of inference completion rate, assignment stability, yield gap, and fallback response delay. The CCP control parameter β plays a critical role in regulating inference task allocation reliability and robustness at runtime, where smaller values enhance the assignment stability and inference success.

In the future, we plan to extend the current orchestration logic to support inter-edge collaboration across multiple edge servers, where CPNs allow the system to elastically reassign workload between XE clusters based on localized resource bottlenecks. This will serve to improve the global inference success rates and system resilience.

Acknowledgments

This research is supported by a grant from the Natural Sciences and Engineering Research Council of Canada (NSERC) under grant number: ALLRP 549919-20, and a grant from Distributive, Ltd. This work was also supported in part by Qatar University under Grant IRCC-2024-494.

References

[1] S. Long, J. Tan, B. Mao, F. Tang *et al.*, "A Survey on Intelligent Network Operations and Performance Optimization Based on Large Language Models," *IEEE Communications Surveys Tutorials*, pp. 1–1, 2025.

[2] H. Du, D. Niyato, J. Kang, Z. Xiong *et al.*, "The Age of Generative AI and AI-Generated Everything," *IEEE Network*, vol. 38, no. 6, pp. 501–512, 2024.

[3] N. Maslej, L. Fattorini, R. Perrault, Y. Gil *et al.* (2025) Artificial Intelligence Index Report 2025. Accessed: 2025-05-16. [Online]. Available: https://hai.stanford.edu/ai-index/2025-ai-index-report

[4] M. Özcan, P. Wiesner, P. Weiß, and O. Kao, "Quantifying the Energy Consumption and Carbon Emissions of LLM Inference via Simulations," *arXiv preprint*, 2025, presented at PECS Workshop, Euro-PAR 2025. [Online]. Available: https://arxiv.org/abs/2507.11417

[5] T. Shen, D. Zhu, Z. Zhao, C. Wu *et al.*, "Will LLMs Scaling Hit the Wall? Breaking Barriers via Distributed Resources on Massive Edge Devices," *arXiv preprint arXiv:2503.08223*, pp. 1–10, 2025.

[6] B. Li, T. Liu, W. Wang, C. Zhao *et al.*, "Agent-as-a-Service: An AI-Native Edge Computing Framework for 6G Networks," *IEEE Network*, vol. 39, no. 2, pp. 44–51, 2025.

[7] X. Tang, C. Cao, Y. Wang, S. Zhang *et al.*, "Computing Power Network: The Architecture Of Convergence Of Computing And Networking Towards 6G Requirement," *China Communications*, vol. 18, no. 2, pp. 175–185, 2021.

[8] S. Long, F. Tang, Y. Li, T. Tan *et al.*, "6G Comprehensive Intelligence: Network Operations and Optimization based on Large Language Models," *IEEE Network*, pp. 1–1, 2024.

[9] Z. Luo, F. Zeng, F. Tang, and N. Kato, "FCSQF: Flexible Traffic Control for ECT-aware Time Sensitive Networking," *IEEE Transactions on Network Science and Engineering*, pp. 1–13, 2025.

[10] B. Mao, J. Liu, Y. Wu, and N. Kato, "Security and privacy on 6g network edge: A survey," *IEEE Communications Surveys Tutorials*, vol. 25, no. 2, pp. 1095–1127, 2023.
[11] L. Zeng, S. Ye, X. Chen, and Y. Yang, "Implementation of Big AI Models for Wireless Networks with Collaborative Edge Computing," *IEEE Wireless Communications*, vol. 31, no. 3, pp. 50–58, 2024.
[12] R. Zhang, J. He, X. Luo, D. Niyato *et al.*, "Toward Democratized Generative AI in Next-Generation Mobile Edge Networks," *IEEE Network*, pp. 1–1, 2025.
[13] S. Yukun, L. Bo, L. Junlin, H. Haonan *et al.*, "Computing Power Network: A Survey," *China Communications*, vol. 21, no. 9, pp. 109–145, 2024.
[14] Y. Chen, C. Jiang, L. Yan, Z. Feng *et al.*, "Research Progress on Computing Power Network for the Power Industry," in *2024 IEEE 12th International Conference on Information, Communication and Networks (ICICN)*, 2024, pp. 201–210.
[15] F. Wang, Z. Zhang, X. Zhang, Z. Wu *et al.*, "A Comprehensive Survey of Small Language Models in the Era of Large Language Models: Techniques, Enhancements, Applications, Collaboration with LLMs, and Trustworthiness," 2024. [Online]. Available: https://arxiv.org/abs/2411.03350
[16] DeepSeek-AI, A. Liu, B. Feng, B. Xue *et al.*, "DeepSeek-V3 Technical Report," 2025. [Online]. Available: https://arxiv.org/abs/2412.19437
[17] Z. Nezami, M. Hafeez, K. Djemame, and S. A. R. Zaidi, "Generative AI on the Edge: Architecture and Performance Evaluation," *arXiv preprint arXiv:2411.17712*, 2024. [Online]. Available: https://arxiv.org/abs/2411.17712
[18] M. Arya and Y. Simmhan, "A Preliminary Performance Analysis of LLM Inference on Edge Accelerators," in *2024 IEEE 31st International Conference on High Performance Computing, Data and Analytics Workshop (HiPCW)*, 2024, pp. 183–184.
[19] I. M. Amer and S. Sorour, "Cost-based Compute Cluster Formation in Edge Computing," in *ICC 2022 - IEEE International Conference on Communications*, 2022, pp. 1611–1616.
[20] R. Kain, S. A. Elsayed, Y. Chen, and H. S. Hassanein, "RUMP: Resource Usage Multi-Step Prediction in Extreme Edge Computing," *Computer Communications*, vol. 210, p. 45–57, Oct. 2023.
[21] R. F. El Khatib, S. A. Elsayed, N. Zorba, and H. S. Hassanein, "Proactive Task Allocation in Extreme Edge Computing for Digital Twin Services," *Internet of Things Journal*, vol. 0, pp. 1–16, 2024.
[22] R. F. El Khatib, S. A. Elsayed, N. Zorba, and H. S. Hassanein, "Optimal Proactive Resource Allocation at the Extreme Edge," in *ICC 2022 - IEEE International Conference on Communications*. IEEE, May 2022.
[23] G. Boulougaris and K. Kolomvatsos, "A QoS-aware, Proactive Tasks Offloading Model for Pervasive Applications," in *2022 9th International Conference on Future Internet of Things and Cloud (FiCloud)*, 2022, pp. 24–31.
[24] J. Liu, N. Liu, L. Liu, S. Li *et al.*, "A Proactive Stable Scheme for Vehicular Collaborative Edge Computing," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 8, pp. 10 724–10 736, 2023.
[25] R. F. El Khatib, N. Zorba, and H. S. Hassanein, "Collaborative Inference in the Extreme Edge: A Proactive Task Allocation Scheme," in *GlobeCom 2025 - IEEE Global Conference on Communications*. IEEE, May 2025.
[26] M. De'bas, S. A. Elsayed, and H. S. Hassanein, "Multitiered Worker-Oriented Resource Allocation at the Extreme Edge," in *GLOBECOM 2022 - 2022 IEEE Global Communications Conference*. IEEE, Dec. 2022.
[27] P. Murthy, A. Mehra, and L. Mishra, "Resource Allocation for Generative AI Workloads: Advanced Cloud Resource Management Strategies for Optimized Model Performance," *Iconic Research and Engineering Journals*, vol. 6, no. 12, pp. 1428–1437, June 2023, paper ID: 1704589.
[28] H. Du, Z. Li, D. Niyato, J. Kang *et al.*, "Enabling AI-Generated Content Services in Wireless Edge Networks," *IEEE Wireless Communications*, vol. 31, no. 3, pp. 226–234, 2024.
[29] G. Qu, Q. Chen, W. Wei, Z. Lin *et al.*, "Mobile Edge Intelligence for Large Language Models: A Contemporary Survey," *IEEE Communications Surveys Tutorials*, pp. 1–1, 2025.
[30] R. Jayanth, N. Gupta, S. Kundu, D. A. Mathaikutty *et al.*, "Towards Real-Time LLM Inference on Heterogeneous Edge Platforms," in *2024 IEEE 31st International Conference on High Performance Computing, Data and Analytics Workshop (HiPCW)*, 2024, pp. 197–198.
[31] Y.-C. Wang, J. Xue, C. Wei, and C. C. J. Kuo, "An overview on generative ai at scale with edge–cloud computing," *IEEE Open Journal of the Communications Society*, vol. 4, pp. 2952–2971, 2023.
[32] M. Xu, H. Du, D. Niyato, J. Kang *et al.*, "Unleashing the Power of Edge-Cloud Generative AI in Mobile Networks: A Survey of AIGC Services," *IEEE Communications Surveys Tutorials*, vol. 26, no. 2, pp. 1127–1170, 2024.
[33] Y. Tian, Z. Zhang, Y. Yang, Z. Chen *et al.*, "An Edge-Cloud Collaboration Framework for Generative AI Service Provision With Synergetic Big Cloud Model and Small Edge Models," *IEEE Network*, vol. 38, no. 5, pp. 37–46, 2024.
[34] Y. Chen, R. Li, Z. Zhao, C. Peng *et al.*, "NetGPT: An AI-Native Network Architecture for Provisioning Beyond Personalized Generative Services," *IEEE Network*, vol. 38, no. 6, pp. 404–413, 2024.
[35] F. Yang, Z. Wang, H. Zhang, Z. Zhu *et al.*, "Efficient Deployment of Large Language Model across Cloud-Device Systems," in *2024 IEEE 37th International System-on-Chip Conference (SOCC)*, 2024, pp. 1–6.
[36] R. Tourani, S. Srikanteswara, S. Misra, R. Chow *et al.*, "Democratizing the edge: A pervasive edge computing framework," *ArXiv*, vol. abs/2007.00641, 2020.
[37] D. Chatzopoulos, C. Bermejo, E. u. Haq, Y. Li *et al.*, "D2d task offloading: A dataset-based q&a," *IEEE Communications Magazine*, vol. 57, no. 2, p. 102–107, Feb. 2019.
[38] H. Bornholdt, K. Röbert, M. Breitbach, M. Fischer *et al.*, "Measuring the edge: A performance evaluation of edge offloading," in *2023 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops)*. IEEE, Mar. 2023.
[39] N. Eshraghi and B. Liang, "Joint offloading decision and resource allocation with uncertain task computing requirement," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, 2019, pp. 1414–1422.
[40] S. Li, C. Li, Y. Huang, B. A. Jalaian *et al.*, "Enhancing resilience in mobile edge computing under processing uncertainty," *IEEE Journal on Selected Areas in Communications*, vol. 41, no. 3, pp. 659–674, 2023.
[41] W. Chang, Y. Xiao, W. Lou, and G. Shou, "Offloading decision in edge computing for continuous applications under uncertainty," *IEEE Transactions on Wireless Communications*, vol. 19, no. 9, pp. 6196–6209, 2020.
[42] T. Ji, C. Luo, L. Yu, Q. Wang *et al.*, "Energy-efficient computation offloading in mobile edge computing systems with uncertainties," *IEEE Transactions on Wireless Communications*, vol. 21, no. 8, pp. 5717–5729, 2022.
[43] D. T. Nguyen, H. T. Nguyen, N. Trieu, and V. K. Bhargava, "Two-Stage Robust Edge Service Placement and Sizing Under Demand Uncertainty," *IEEE Internet of Things Journal*, vol. 9, no. 2, pp. 1560–1574, 2022.
[44] P. Qin, Y. Fu, G. Tang, X. Zhao *et al.*, "Learning Based Energy Efficient Task Offloading for Vehicular Collaborative Edge Computing," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 8, pp. 8398–8413, 2022.
[45] R.-J. Reifert, H. Dahrouj, A. A. Ahmad, H. Gacanin *et al.*, "Robust Communication and Computation using Deep Learning via Joint Uncertainty Injection," in *2024 19th International Symposium on Wireless Communication Systems (ISWCS)*, 2024, pp. 1–6.
[46] P. Lai, Q. He, X. Xia, F. Chen *et al.*, "Dynamic User Allocation in Stochastic Mobile Edge Computing Systems," *IEEE Transactions on Services Computing*, vol. 15, no. 5, pp. 2699–2712, 2022.
[47] P. A. Apostolopoulos, G. Fragkos, E. E. Tsiropoulou, and S. Papavassiliou, "Data offloading in uav-assisted multi-access edge computing systems under resource uncertainty," *IEEE Transactions on Mobile Computing*, vol. 22, no. 1, pp. 175–190, 2023.
[48] S. Xia, Z. Yao, Y. Li, Z. Xing *et al.*, "Distributed Computing and Networking Coordination for Task Offloading Under Uncertainties," *IEEE Transactions on Mobile Computing*, vol. 23, no. 5, p. 5280–5294, May 2024.
[49] X. Zhang, J. Zhang, Z. Liu, Q. Cui *et al.*, "MDP-Based Task Offloading for Vehicular Edge Computing Under Certain and Uncertain Transition Probabilities," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 3, pp. 3296–3309, 2020.
[50] A. Karanika, P. Oikonomou, K. Kolomvatsos, and T. Loukopoulos, "A Demand-driven, Proactive Tasks Management Model at the Edge," in *2020 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, 2020, pp. 1–8.
[51] GSMA Intelligence. "Edge Computing Coming to a Place Near You". [Online]. Available: https://www.gsmaintelligence.com/research/edge-compute-coming-to-a-place-near-you

[52] J. Liang, B. Ma, Z. Feng, and J. Huang, "Reliability-Aware Task Processing and Offloading for Data-Intensive Applications in Edge Computing," *IEEE Transactions on Network and Service Management*, vol. 20, no. 4, pp. 4668–4680, 2023.

[53] J. Xie, Y. Jia, W. Wen, Z. Chen *et al.*, "Dynamic d2d multihop offloading in multi-access edge computing from the perspective of learning theory in games," *IEEE Transactions on Network and Service Management*, vol. 20, no. 1, pp. 305–318, 2023.

[54] W. Jiang, D. Feng, Y. Sun, G. Feng *et al.*, "Joint computation offloading and resource allocation for d2d-assisted mobile edge computing," *IEEE Transactions on Services Computing*, vol. 16, no. 3, pp. 1949–1963, 2023.

[55] T. Fang, F. Yuan, L. Ao, and J. Chen, "Joint task offloading, d2d pairing, and resource allocation in device-enhanced mec: A potential game approach," *IEEE Internet of Things Journal*, vol. 9, no. 5, pp. 3226–3237, 2022.

[56] A. Khazali, A. Bozorgchenani, D. Tarchi, M. G. Shayesteh *et al.*, "Joint task assignment, power allocation and node grouping for cooperative computing in noma-mmwave mobile edge computing," *IEEE Access*, vol. 11, p. 93664–93678, 2023.

[57] M. Kantardjian, S. A. Elsayed, and H. S. Hassanein, "Dynamic worker availability prediction at the extreme edge," in *GLOBECOM 2023 - 2023 IEEE Global Communications Conference*. IEEE, Dec. 2023.

[58] H. Flores, A. Zuniga, S. Tarkoma, L. Tonetto *et al.*, "Collaboration stability: Quantifying the success and failure of opportunistic collaboration," *Computer*, vol. 55, no. 8, p. 70–81, Aug. 2022.

[59] J. R. Birge and F. Louveaux, *Introduction to Stochastic Programming*, 2nd ed., ser. Springer Series in Operations Research and Financial Engineering. New York: Springer, 2011.

[60] S. Ahmed and A. Shapiro, *Solving Chance-Constrained Stochastic Programs via Sampling and Integer Programming*. INFORMS, Sep. 2008, p. 261–269.

[61] S. Ahmed and A. Shapiro, *The Sample Average Approximation Method for Stochastic Programs with Integer Recourse*. INFORMS, Sep. 2002, p. 261–269.

[62] S. Boyd, N. Parikh, E. Chu, B. Peleato *et al.*, "Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers," in *Foundations and Trends in Machine Learning*. Now Publishers, Inc., 2011, vol. 3, no. 1, pp. 1–122.

[63] S. Arpón, T. H. de Mello, and B. K. Pagnoncelli, "An ADMM Algorithm for Two-stage Stochastic Programming Problems," *Annals of Operations Research*, vol. 286, no. 1-2, pp. 559–582, 2020.

[64] N. Mohammed and M. M. Ali, "Optimal Sizing of Hybrid Renewable Energy System Using Two-Stage Stochastic Programming," *Mathematical Problems in Engineering*, p. Article ID 2361858, 2024.

[65] D. Phan and S. Ghosh, "Two-Stage Stochastic Optimization For Optimal Power Flow Under Renewable Generation Uncertainty," *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, vol. 24, no. 1, pp. 1–22, 2014.

[66] R. F. El Khatib, N. Zorba, and H. S. Hassanein, "Rapid Sensing-based Emergency Detection: A Sequential Approach," *Computer Communications*, vol. 159, pp. 222–230, Jun. 2020.

[67] "NVIDIA Triton Inference Server," https://developer.nvidia.com/nvidia-triton-inference-server, 2023.

[68] S. Ye, B. Ouyang, L. Zeng, T. Qian *et al.*, "Jupiter: Fast and Resource-Efficient Collaborative Inference of Generative LLMs on Edge Devices," *arXiv preprint arXiv:2504.08242*, pp. 1–15, 2025. [Online]. Available: https://arxiv.org/abs/2504.08242

[69] D. Rivkin, F. Hogan, A. Feriani, A. Konar *et al.*, "AIoT Smart Home via Autonomous LLM Agents," *IEEE Internet of Things Journal*, vol. 12, no. 3, pp. 2458–2472, 2025.

[70] E. King, H. Yu, S. Lee, and C. Julien, "Sasha: Creative Goal-Oriented Reasoning in Smart Homes with Large Language Models," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 8, no. 1, pp. 1–38, 2024.

[71] J. Li, H. Lv, B. Lei, and Y. Xie, "A Computing Power Resource Modeling Approach for Computing Power Network," in *2022 International Conference on Computer Communications and Networks (ICCCN)*, 2022, pp. 1–2.

[72] B. Lei, Q. Zhao, and J. Mei, "Computing Power Network: An Interworking Architecture of Computing and Network Based on IP Extension," in *2021 IEEE 22nd International Conference on High Performance Switching and Routing (HPSR)*, 2021, pp. 1–6.

[73] Y. Cui, W. Tong, T. Liu, K. Cao *et al.*, "Energy-Aware Incentive Mechanism for Hierarchical Federated Learning Using Water Filling Technique," *IEEE Transactions on Industrial Informatics*, vol. 20, no. 12, pp. 14 214–14 225, 2024.

[74] A. O. S. Project, "Power Profiles and Energy Reporting," https://source.android.com/docs/core/power/values, 2023, accessed: 2024-05-15.

[75] J. Luedtke, S. Ahmed, and G. L. Nemhauser, "An integer programming approach for linear programs with probabilistic constraints," *Mathematical Programming*, vol. 122, no. 2, p. 247–272, Oct. 2008.

[76] G. L. Nemhauser and L. A. Wolsey, *Integer and Combinatorial Optimization*. New York: Wiley, 1988.

[77] A. Schrijver, *Theory of Linear and Integer Programming*, ser. Wiley-Interscience Series in Discrete Mathematics and Optimization. Chichester, UK: John Wiley & Sons, 1998.

[78] B. He, H. Yang, and S. Wang, "Alternating Direction Method with Self-adaptive Penalty Parameters for Monotone Variational Inequalities," *Journal of Optimization Theory and Applications*, vol. 106, no. 2, pp. 337–356, 2000.

[79] Z. Xie, Y. Xu, H. Xu, Y. Liao *et al.*, "A Novel Hat-Shaped Device-Cloud Collaborative Inference Framework for Large Language Models," *arXiv preprint arXiv:2503.18989*, 2025.

[80] D. C. Nguyen, P. N. Pathirana, M. Ding, and A. Seneviratne, "Secure computation offloading in blockchain based iot networks with deep reinforcement learning," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 4, pp. 3192–3208, 2021.