Carbon-Aware Edge Computing for Internet of Everything Networks: A Digital Twin Approach

Dang Van Huynh, *Member, IEEE,* Saeed R. Khosravirad, *Senior Member, IEEE,* Vishal Sharma, *Senior Member, IEEE,* Joongheon Kim, *Senior Member, IEEE,* Berk Canberk, *Senior Member, IEEE,* and Trung Q. Duong, *Fellow, IEEE*

***Abstract*—The rapid growth of edge computing has enabled low-latency and high-efficiency processing for a wide range of applications; however, it also leads to significant energy consumption and carbon emissions. In this context, this study investigates a CO_2 emission minimisation problem in a digital twin-aided edge computing system, aiming to optimise task offloading decisions, transmit power, and processing rates of Internet of Things (IoT) devices. To address the formulated mixed-integer nonlinear programming problem, we propose two solutions: an alternating optimisation method based on the successive convex approximation framework and a deep reinforcement learning (DRL) approach. Extensive simulations validate the effectiveness of the proposed solutions, demonstrating significant reductions in CO_2 emissions, robust optimisation performance, and superior results compared to benchmark schemes. The findings highlight the feasibility of integrating advanced optimisation and artificial intelligence-driven techniques to achieve environmentally sustainable and high-performance edge computing systems, paving the way for greener technological innovation.**

***Index Terms*—Digital twin, carbon neutrality, deep reinforcement learning, mobile edge computing, sustainability.**

I. Introduction

Unlike traditional computing architectures that often require data to be transmitted to distant data centres for processing, mobile edge computing (MEC) shifts computational power closer to the data's source [2]. This proximity enables significantly faster processing times and reduced latency, making MEC ideal for latency-sensitive applications such as smart factories, intelligent transportation systems, and immersive technologies [3], [4]. By processing data locally or near the edge, MEC not only enhances performance but also reduces the bandwidth requirements associated with data transfer to centralised locations. Despite its advantages, MEC introduces a range of technical challenges that demand innovative solutions [5]. These challenges include efficient resource allocation to optimise performance within the constraints of limited computational and energy resources, seamless integration with diverse and existing network infrastructures, and the development of robust security measures to protect sensitive information from potential threats [6]. Addressing these complexities is critical to unlocking MEC's full potential and enabling its widespread adoption across various domains.

Recently, digital twin (DT) technology has emerged as a pivotal enabler for edge computing, facilitating the development of a new generation of real-time networking systems. With the capability to create comprehensive digital replicas of physical systems, DT provides the foundation for making prompt and optimal decisions to control and manage these systems efficiently [7]–[9]. Through DT synchronisation mechanisms, real-time data from physical MEC systems—including computational task descriptions, the operational status of user devices and edge servers, and channel conditions—can be accessed and processed seamlessly to deliver optimal configurations. Leveraging the DT concept, numerous studies have proposed optimal designs for edge computing systems by addressing critical challenges such as adaptive computing adjustments, power control optimisation, service placement, and user association [10]–[12]. These advancements significantly contribute to the development of transformative applications that demand low latency, high reliability, and dynamic adaptability. Examples include the metaverse, intelligent transportation systems, and autonomous factories. By enabling such applications, DT-assisted edge computing demonstrates its potential to redefine the boundaries of technological innovation in a highly interconnected world.

Notably, recent studies in edge computing have focused on carbon-aware problems, exploring the trade-off between meeting stringent user requirements—such as low latency and ultra-fast processing—and minimising environmental impacts [13]–[15]. These significant efforts are making positive strides

D. V. Huynh is with the Faculty of Engineering and Applied Science, Memorial University, St. John's, NL A1B 3X5, Canada (e-mail: vdhuynh@mun.ca).

S. R. Khosravirad is with Nokia Bell Labs, Murray Hill, NJ 07964 USA (e-mail: saeed.khosravirad@nokia-bell-labs.com).

V. Sharma is with the School of Electronics, Electrical Engineering and Computer Science, Queen's University Belfast, Belfast, BT7 1NN, UK (email: v.sharma@qub.ac.uk).

J. Kim is with the Department of Electrical and Computer Engineering, Korea University, Seoul 02841, South Korea (e-mail: joongheon@korea.ac.kr).

B. Canberk is with the School of Computing, Engineering and The Built Environment at Edinburgh Napier University, Edinburgh, EH11 4BN, UK (e-mail: b.canberk@napier.ac.uk).

T. Q. Duong is with the Faculty of Engineering and Applied Science, Memorial University, St. John's, NL A1C 5S7, Canada and also with the School of Electronics, Electrical Engineering and Computer Science, Queen's University Belfast, BT7 1NN Belfast, U.K., (e-mail: tduong@mun.ca).

This work was supported in part by the Canada Excellence Research Chair (CERC) Programm, project number CERC-2022-00109. The work of B. Canberk is supported in part by The Scientific and Technological Research Council of Turkey (TUBITAK) Frontier R&D Laboratories Support Program for BTS Advanced AI Hub: BTS Autonomous Networks and Data Innovation Lab Project 5239903. The work of V. Sharma and T. Q. Duong was supported in part by UKRI in the UKRI-Horizon Europe program with the UKRI reference number 10061165 under MISO project "Autonomous Multi-Format In-Situ Observation Platform for Atmospheric Carbon Dioxide and Methane Monitoring in Permafrost & Wetlands".

This paper was presented in part at IEEE Global Communications Conference 8–12 December 2024, Cape Town, South Africa [1].

Corresponding authors is Trung Q. Duong.

toward addressing the critical global issue of climate change through technological innovation. To tackle this challenge, numerous efficient solutions have been proposed, including optimising computing resources, managing transmission power, and enhancing resource utilisation to minimise energy consumption and reduce CO_2 emissions. Research in this area is still in its early stages, and given the urgency of climate change, there are numerous open challenges and opportunities for meaningful contributions. Advancing this field can lead to environmentally sustainable solutions that benefit the planet while simultaneously driving technological progress. Such contributions hold the potential to strike a balance between environmental preservation and the continuous momentum of technological innovation.

A. Literature Review

In MEC, the optimal design of joint communication and computing resource allocation stands out as a primary research focus [16]–[18]. In particular, a multi-UAV employing MEC was considered in [16] to enable the requirement of ultra-reliable low latency communications (URLLCs) in intelligent autonomous transport applications, where the energy consumption of the system was minimised by jointly optimizing communication and computation parameters. In [17], a scheme for joint task offloading and resource allocation aimed at minimising total processing delay in Internet of vehicle (IoV) systems was proposed. The scheme optimised task scheduling, channel allocation, and computing resource allocation for the vehicles, aiming to enhance overall system efficiency. Additionally, a solution for distributed resource optimisation was introduced in [18] to address fairness-aware latency minimisation among users in MEC systems assisted by digital twin (DT) technology, which optimised various communication and computation variables, such as transmit power, bandwidth allocation, task offloading portions, and processing rates of user equipment, through both centralised and distributed optimisation approaches. Overall, the research focus on resource allocation in MEC systems has garnered attention due to various technical challenges. However, there are still open issues to explore in these areas, including understanding the environmental impact of communication and computing systems, managing the trade-offs between system budget, computing capacity, quality-of-service (QoS), quality-of-experience (QoE), and reducing the carbon footprint released into the environment.

Regarding DT technology, recent research has increasingly focused on leveraging DT technology to address energy consumption challenges in edge computing environments [19]–[21]. A DT-assisted intelligent partial offloading approach for vehicular edge computing has been proposed, optimising resource allocation and reducing energy consumption in vehicular networks by creating a virtual representation of the system to facilitate efficient decision-making [19]. A DT-assisted edge computation offloading framework for industrial Internet of Things (IIoT) systems has been developed, integrating non-orthogonal multiple access (NOMA) to minimise task completeion delay [20]. A DT-enabled continual learning strategy for service provisioning in edge computing has been introduced, enabling dynamic adaptation to resource demands while maximising total utility gain [21]. These advancements highlight the potential of DTs in providing accurate system modelling and real-time insights, paving the way for energy-optimised edge computing solutions across diverse applications.

In the realm of sustainable computing, carbon neutralisation has emerged as a critical focus in the design and operation of edge computing systems. Recent advancements have introduced innovative approaches to address these challenges [13]–[15], [22]–[28]. These studies emphasise efficient solutions such as advanced resource scheduling algorithms, energy-aware task offloading, and collaborative energy management techniques to optimise resource utilisation and minimise carbon footprints. For instance, a sustainable resource management framework in [13] utilised deep reinforcement learning (DRL) models to significantly reduce energy consumption and CO_2 emissions. Similarly, [22] introduced an optimal task scheduling and offloading solution, reformulating a mixed-integer linear programming problem using graph-based techniques to address carbon footprint reduction in edge computing.

Research has also tackled more intricate challenges. One such challenge is the joint machine learning (ML) task offloading and carbon emission rights purchasing problem, as discussed in [23], where a two-timescale Lyapunov optimisation technique has been employed to achieve optimal decision-making. Additionally, online collaborative energy-network resource scheduling has been proposed for wireless power transfer (WPT)-enabled green edge computing that effectively balances energy and network resources to achieve carbon-neutral operations [15]. Furthermore, [25] presented a Lyapunov optimisation-based approach to reduce carbon emissions for computation-intensive tasks in queuing-aware network models. Joint task offloading and energy-sharing mechanisms have also been explored as effective strategies for reducing the carbon footprint of edge computing. Such approaches have demonstrated considerable energy savings and environmental impact reductions [26]. Expanding on this, electric vehicles (EVs) have been integrated into edge data centres as mobile energy sources, providing flexible energy support during peak demand periods and further enhancing sustainability [14].

AI-powered sustainable resource management frameworks have demonstrated significant potential in serverless edge computing environments. These frameworks leverage advanced machine learning algorithms to dynamically optimise resource allocation, predict workload patterns, and adapt to fluctuating energy availability, enabling significant reductions in energy consumption and operational costs. For example, ATOM focuses on dynamic resource allocation to optimise energy efficiency while maintaining system performance [13]. Likewise, [24] proposed a DRL-based strategy for edge computing management, aimed at minimising long-term operational costs and promoting low-carbon solutions. FaasHouse, introduced in [27], integrates energy-aware resource scheduling to offer a sustainable solution. By optimising resource allocation based

on energy availability and workload demands, it enhances operational efficiency. Similarly, EESaver dynamically adjusts resource usage to minimise energy consumption without compromising service quality [28]. These systems underline the importance of intelligent resource management in advancing energy-efficient operations.

Collectively, these advancements highlight the importance of integrating sustainability principles into the edge computing paradigm. By combining techniques such as intelligent resource scheduling, energy-aware task management, and collaborative energy sharing, these solutions pave the way for environmentally responsible and highly energy-efficient systems. Importantly, these studies have effectively addressed the pressing challenges of carbon reduction in modern computing infrastructures. Overall, sustainable edge computing represents a significant research direction with direct implications for a global issue—reducing carbon footprint in advanced computing systems. Recent efforts in this area have been directed towards optimal designs of resource allocation solutions to minimise CO_2 emissions effectively.

B. Motivation and Contributions

The increasing demand for computational resources in edge computing, driven by latency-sensitive applications such as intelligent transportation, industrial automation, and immersive technologies, has exacerbated energy consumption and CO_2 emissions. While recent advancements in resource optimisation and energy-aware solutions have achieved notable progress, significant challenges remain in effectively balancing the trade-off between performance requirements and environmental sustainability. This challenge is particularly critical as addressing climate change necessitates solutions that both minimise carbon footprints and meet the growing performance demands of modern edge systems. Moreover, emerging technologies such as digital twins and DRL offer new opportunities to design intelligent, adaptive strategies that can optimise communication and computing resources while reducing energy consumption. Therefore, there is a pressing need to develop innovative, practical solutions to achieve CO_2 emission minimisation in edge computing systems without compromising user experience. This study is motivated by the potential to harness advanced techniques, such as DRL, optimisation, and digital twins, to contribute to a sustainable edge computing ecosystem, enabling the development of energy-efficient, environmentally responsible, and high-performance systems.

Main contributions of this paper can be summarised as follows:

- We formulate a DT-enabled CO_2 minimisation problem within the edge computing framework, incorporating constraints such as delay tolerance, users' energy budgets, computing capacity of edge servers, and quality-of-service (QoS) transmission rates. The formulated problem highlights the critical trade-off between performance requirements and environmental sustainability in future edge computing systems.
- To address the formulated problem, we propose an alternating optimisation approach based on the successive convex approximation (SCA) framework. This approach efficiently handles optimisation variables, including transmission power, processing rates, and offloading decisions, to achieve near-optimal solutions.
- In addition to the SCA-based method, we introduce a DRL-based solution that leverages the ability of DRL agents to learn optimal policies through interaction with the dynamic and uncertain environment of DT-enabled edge computing systems. Through continuous training and feedback, the DRL framework effectively optimises resource allocation and task offloading strategies to minimise CO_2 emissions while maintaining system performance.
- Extensive simulations are performed to validate the effectiveness of the proposed solutions. The numerical results demonstrate the performance of the optimisation process, highlight the superiority of the proposed solutions compared to benchmark schemes, and analyse the impacts of key system parameters on CO_2 emissions and system performance.

C. Paper Structure and Notations

The remainder of this paper is organised as follows. Section II presents the system model and problem formulation, detailing the studied model and the formulation of the CO_2 minimisation problem. Section III develops the SCA-based method to efficiently solve the formulated optimisation problem. In Section IV, we introduce the DRL-based solution, including the reinforcement learning transformation, the proposed algorithm, and its implementation details. Section V provides extensive simulation results, demonstrating the effectiveness of the proposed solutions, comparing their performance with benchmark schemes, and analysing the impact of key system parameters. Finally, Section VI summarises the key findings of this paper and discusses potential future research directions.

Throughout the paper, scalar values are denoted using regular lowercase letters, while vector notations are represented using bold lowercase letters. Variables or parameters associated with the m-th user device are indexed by the subscript m. The i-th iteration in the iterative optimisation process is represented with the superscript (i). The notation $|\cdot|$ denotes the Euclidean norm of a vector, and $\mathbb{C}$ represents the set of complex numbers. The term $\nabla_{\theta^\mu} J$ indicates the gradient of the objective function J with respect to the parameters θ^μ. Additionally, $\mathbb{E}_{s\sim p^\pi}$ denotes the expectation taken over the states s, sampled from the state distribution p^π. These notations are consistently used throughout the paper to ensure clarity and precision in the mathematical formulations.

II. System Model and Problem Formulation

In this paper, we consider a single base station, which is equipped with multiple antennas ($L > 0$) and associated with an edge server to process the offloaded tasks from IoT devices (UEs). There are M single-antenna IoT devices in the system denoted by the set of $\mathcal{M} = \{1, ..., M\}$. An illustration of the considered system model is displayed as Fig. 1. DT technology

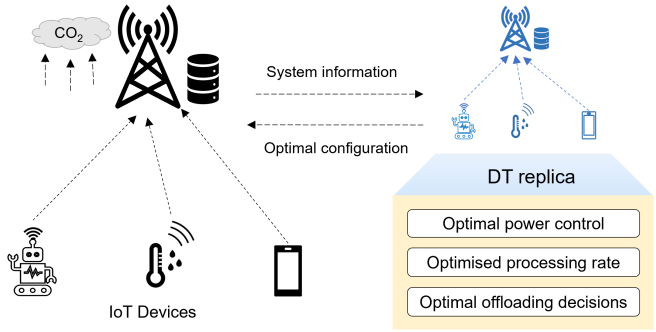

Fig. 1. An illustration of the DT-enabled carbon-aware edge computing systems (DTCAEC).

is employed to create a virtual replica of the physical system. With real-time data updates and remote control capabilities provided by DT, the system can be managed efficiently and sustainably through optimal designs of transmit power, computing rate, and task offloading decisions.

A computational task generated at the m-th UE is modelled as $J_m = (S_m, C_m, D_m^{\max})$, where S_m is the size of the task, C_m is the required CPU cycles to process the task, and $D_m^{\max}$ is the delay tolerance of the task. Due to limitations in terms of computing capacity and energy budget of constrained IoT devices, the m-th UE has to decide whether to offload the task to MEC or not, which is modelled as a binary variable $\alpha_m = \{0, 1\}$. Specially, when $\alpha_m = 1$, the computation task is offloaded to the MEC; otherwise, it is processed locally at the m-th IoT devices.

A. Representations of DT-enabled Edge Computing Systems

DT has recently emerged as a promising technology to harness the potential of MEC systems, garnering significant attention from researchers in the field [8], [10], [29]. DT can be effectively utilised in MEC-based systems by virtually representing the computing capacity of physical devices, such as IoT devices and edge servers, enabling optimal decision-making for efficient control and management. In this paper, we adopt the assumption that DT is leveraged to comprehensively control and manage the entire physical system. To achieve this, the DT-enabled edge server is modelled as follows: $\mathrm{DT}_0 = (f_m^0, \hat{f}_m^0, \alpha_m)$, where f_m^0 (cycles/second) is the estimated processing rate of the edge server at DT, $\hat{f}_0$ is the deviation between the estimate and the real value of MEC's processing rate [8], [10], [18].

Regarding the m-th UE, its DT model is given by $\mathrm{DT}_m = (f_m, \hat{f}_m, p_m, \alpha_m)$, where $f_m, \hat{f}_m$, and p_m are the estimated processing rate, the deviation value of the processing rate, and the transmission power of UE, respectively. We assume that the necessary infrastructure is fully implemented to facilitate real-time interactions between the DT model and the physical system for data collection and remote control [30].

B. Wireless Transmission Model

Wireless connections between the base station (BS) and the m-th UE are established to facilitate communication. The channel vector $\mathbf{h}_m \in \mathbb{C}^{L\times 1}$ represents the connection between the BS and the m-th UE and is expressed as $\mathbf{h}_m = \sqrt{g_m}\bar{\mathbf{h}}m$, where g_m encompasses the large-scale channel coefficient, accounting for factors like pathloss and shadowing, while $\bar{\mathbf{h}}_m$ follows a small-scale fading distribution of $\mathcal{CN}(0, \mathbf{I})$. The received signal vector at the BS is represented as an $L \times 1$ matrix, expressed as

$$\mathbf{y} = \sum_{m=1}^{M} \mathbf{h}_m \sqrt{p_m} s_m + \mathbf{n}_k. \tag{1}$$

Here, p_m denotes the transmission power of the m-th device, s_m represents the zero-mean and unit-variance Gaussian information message from the m-th UE, and $\mathbf{n}_k \sim \mathcal{CN}(\mathbf{0}, N_0\mathbf{I}_L)$ signifies the additive white Gaussian noise (AWGN) encountered during data transmission, where N_0 denotes the noise power. As a result, the transmission rate (bit/s) of wireless transmissions is calculated as

$$R_m(\mathbf{p}) = B \log_2 \left(1 + \frac{p_m \|\mathbf{h}_m\|^2}{\mathcal{I}_m(\mathbf{p}) + N_0}\right) \tag{2}$$

where $\mathcal{I}_m(\mathbf{p}) = \sum_{n\neq m}^{M} \frac{p_n |\mathbf{h}_m^H \mathbf{h}_n|^2}{\|\mathbf{h}_m\|^2}$ is the interference power.

Then, the transmission delay (seconds) for task offloading from the m-th UE to the edge server is calculated as

$$D_m^{\mathrm{tx}}(\mathbf{p}) = \frac{\alpha_m S_m}{R_m(\mathbf{p})}. \tag{3}$$

C. Processing Delay Models

The processing delay of the computational task at the m-th UE is determined by the offloading decision, the required CPU cycles of the task, and the actual processing rate of the IoT device. This delay can be expressed as [8]:

$$D_m^{\mathrm{IoT}}(\alpha_m, f_m) = \frac{(1-\alpha_m) C_m}{f_m + \hat{f}_m}. \tag{4}$$

Similarly, the processing delay of the task offloaded to the edge server is expressed as

$$D_m^{\mathrm{MEC}}(\alpha_m, f_m^0) = \frac{\alpha_m C_m}{f_m^0 + \hat{f}_m^0}. \tag{5}$$

Consequently, the end-to-end (e2e) delay of the offloaded task from the m-th UE consists of the local processing delay, wireless transmission delay, and the edge processing delay, defined as

$$D_m(\alpha_m, \mathbf{p}, \mathbf{f}) = D_m^{\mathrm{IoT}}(\alpha_m, f_m) + D_m^{\mathrm{tx}}(\mathbf{p}) + D_m^{\mathrm{MEC}}(\alpha_m, f_m^0). \tag{6}$$

D. Energy Consumption Models

To execute local processing and task offloading through wireless connections, IoT devices consume energy for these operations. Hence, the total energy consumption (measured in joules) of the m-th UE comprises the energy expended on local processing (E_m^{cp}) and the wireless transmission (E_m^{cm}), expressed as:

$$\begin{aligned} E_m(\alpha_m, \mathbf{p}, \mathbf{f}) &= E_m^{\mathsf{cp}}(\alpha_m, f_m) + E_m^{\mathsf{cm}}(\alpha_m, \mathbf{p}) \\ &= \theta_m (1-\alpha_m) C_m (f_m + \hat{f}_m)^2 + \frac{p_m \alpha_m S_m}{R_m(\mathbf{p})}, \end{aligned} \tag{7}$$

where θ_m is the parameter for computation energy consumption of the m-th IoT (Watt.s^3/cycle^3).

Similarly, the energy consumption for computation at the MEC is given by

$$E_m^0(\alpha_m, \mathbf{f}) = \theta_0 \alpha_m C_m (f_m^0 + \hat{f}_m^0)^2, \tag{8}$$

where θ_0 is the parameter for computation energy consumption of the MEC server.

E. CO_2 Emissions Model

As IoT devices and edge servers actively participate in computing and communication tasks, the energy expended during these operations translates into CO_2 emissions, contributing to environmental impact. The environmental impact of computing and communication technologies is multifaceted, encompassing not only the direct emissions from energy usage but also the indirect impacts stemming from manufacturing, infrastructure, and electronic waste. Deriving the carbon emission amount involves navigating through various complex processes, yet a broad estimation can be attained by [13].

$$\xi_m(\alpha_m, \mathbf{p}, \mathbf{f}) = \eta C_{\mathrm{IE}} \Big[E_m(\alpha_m, \mathbf{p}, \mathbf{f}) + E_m^0(\alpha_m, \mathbf{f}) \Big], \tag{9}$$

where ξ_m is carbon emission (kg CO_2), C_{IE} is carbon intensity of electricity (kgCO_2/kWh), and η is used for energy conversion. The amount of C_{IE} differs across regions, and for the purposes of this paper, it stands at 182 gCO_2/kWh for the London area [13].

F. Optimisation Problem Formulation

This paper focuses on minimising the maximum potential CO_2 emissions associated with processing computational tasks within the network, taking into account specified delay requirements, energy constraints for IoT devices, and the computing capacity of the edge server. Consequently, the optimisation problem addressed in this study is formulated as follows.

$$\min_{\boldsymbol{\alpha}, \mathbf{p}, \mathbf{f}} \ \max_{\forall m} \{\xi_m(\boldsymbol{\alpha}, \mathbf{p}, \mathbf{f})\}, \tag{10a}$$

$$\text{s.t. } D_m(\alpha_m, \mathbf{p}, \mathbf{f}) \le D_m^{\max}, \forall m \tag{10b}$$

$$E_m(\alpha_m, \mathbf{p}, \mathbf{f}) \le E_m^{\max}, \forall m, \tag{10c}$$

$$R_m(\mathbf{p}) \ge R_m^{\min}, \forall m, \tag{10d}$$

$$\sum_{m=1}^{M} \alpha_m f_m^0 \le F_{\max}, \tag{10e}$$

$$\alpha_m \in \{0, 1\}, \forall m. \tag{10f}$$

As outlined in (10), constraint (10b) represents the delay requirement for each computational task. Constraints (10c) and (10e) define the energy budget allocated to IoT devices and the computing capacity available at the edge server, respectively. The quality of service (QoS) requirement for the wireless transmission link is specified in constraint (10d). Lastly, constraint (10f) pertains to the binary decision regarding task offloading.

III. Proposed SCA-based Optimisation Solution

The problem (10) is evidently a mixed-integer nonlinear programming (MINLP) problem, posing significant computational challenges for direct solution. Compounding the complexity are the strong coupling between binary and continuous variables, exemplified by $E_m(\alpha_m, \mathbf{p}, \mathbf{f})$, $T_m(\alpha_m, \mathbf{p}, \mathbf{f})$, and non-convex constraints such as (10b) and (10c). To address these complexities, we introduce an alternating optimisation approach based on the SCA framework tailored to tackle this challenging problem [31]. In order to solve the problem (10) with the alternating approach, we consider three subproblems, including optimal transmit power control, optimal the estimated processing rate, and optimal task offloading decisions. The subsequent subsections detail the development of our proposed solution.

A. Optimal Transmit Power Control

To begin, we address the optimal transmit power problem. In order to find the most efficient power control for wireless transmissions from the m-th IoT devices to the BS, we establish the following optimisation problem.

$$\min_{\mathbf{p} | \mathbf{f}^{(i)}, \boldsymbol{\alpha}^{(i)}} \ \max_{\forall m} \{\xi_m(\boldsymbol{\alpha}^{(i)}, \mathbf{p}, \mathbf{f}^{(i)})\}, \tag{11a}$$

$$\text{s.t. } D_m(\alpha_m^{(i)}, \mathbf{p}, \mathbf{f}^{(i)}) \le D_m^{\max}, \forall m \tag{11b}$$

$$E_m(\alpha_m^{(i)}, \mathbf{p}, \mathbf{f}^{(i)}) \le E_m^{\max}, \forall m, \tag{11c}$$

$$R_m(\mathbf{p}) \ge R_m^{\min}, \forall m. \tag{11d}$$

As we can see from (11), the objective function and constraints (11b), (11c), (11d) are non-convex. Therefore, we process these constraints by convexifying the transmission rate function, the delay and the energy expressions. Firstly, we address the non-convex transmission rate function because it is the main component of transmission latency as well as energy computation of communications. To do this, we apply the following inequality [32], [33]:

$$\ln(1 + \frac{x}{y}) \ge u - \frac{v}{x} - wy, \tag{12}$$

where $u = \ln\left(1 + \frac{\bar{x}}{\bar{y}}\right) + 2\frac{\bar{x}}{\bar{x}+\bar{y}} > 0$, $v = \frac{\bar{x}^2}{\bar{x}+\bar{y}} > 0$, and $w = \frac{\bar{x}}{(\bar{x}+\bar{y})\bar{y}} > 0$. with $x > 0, y >$, and $(\bar{x}, \bar{y})$ are the feasible point of (x, y) into (2) with $x = p_m \|\mathbf{h}_m\|^2$, $\bar{x} = p_m^{(i)} \|\mathbf{h}_m\|^2$, $y = \mathcal{I}_m(\mathbf{p}) + N_0$, and $\bar{y} = \mathcal{I}_m(\mathbf{p}^{(i)}) + N_0$. As a results, the transmission rate $R_m(\mathbf{p})$ can be approximated as follows

$$R_m(\mathbf{p}) \ge \frac{B}{\ln 2} \big[u - \frac{v}{x} - wy \big] \triangleq R_m^{(i)}(\mathbf{p}^{(i)}), \tag{13}$$

where

$$u = \ln\left(1 + \frac{p_m^{(i)} \|\mathbf{h}_m\|^2}{\mathcal{I}_m(\mathbf{p}^{(i)}) + N_0}\right) + \frac{2 p_m^{(i)} \|\mathbf{h}_m\|^2}{p_m^{(i)} \|\mathbf{h}_m\|^2 + \mathcal{I}_m(\mathbf{p}^{(i)}) + N_0},$$

$$v = \frac{\left(p_m^{(i)} \|\mathbf{h}_m\|^2\right)^2}{p_m^{(i)} \|\mathbf{h}_m\|^2 + \mathcal{I}_m(\mathbf{p}^{(i)}) + N_0},$$

$$w = \frac{p_m^{(i)} \|\mathbf{h}_m\|^2}{(p_m^{(i)} \|\mathbf{h}_m\|^2 + \mathcal{I}_m(\mathbf{p}^{(i)}) + N_0)\big(\mathcal{I}_m(\mathbf{p}^{(i)}) + N_0\big)}.$$

Consequently, the constraint (11d) is now equivalent to the

following constraint

$$R_m^{(i)}(\mathbf{p}^{(i)}) \geq R_m^{\min}, \forall m, i. \tag{14}$$

To deal with (11b), we introduce variables $\mathbf{r} = \{r_m\}_{\forall m} \geq 1/R_m^{(i)}(\mathbf{p}^{(i)})$. At the i-th iteration, the constraint (11b) can be expressed as follows.

$$\frac{(1-\alpha_m^{(i)})C_m}{f_m^{(i)} + \hat{f}_m} + \alpha_m S_m r_m + \frac{\alpha_m^{(i)} C_m}{f_m^{0(i)} + \hat{f}_m^0} \leq D_m^{\max}, \tag{15}$$

which is a convex constraint under the variables of $\mathbf{p}$.

Similarly, we apply the introduced variables $\mathbf{r}$ to approximate $E_m(\alpha_m^{(i)}, \mathbf{p}, \mathbf{f}^{(i)})$ in (10c) as follows.

$$\theta_m(1-\alpha_m^{(i)})C_m(f_m^{(i)} + + \hat{f}_m)^2 + p_m r_m \alpha_m^{(i)} S_m \leq E_m^{\max}. \tag{16}$$

However, (16) is still not a convex constraint. Therefore, we apply the following inequality to convexify (16).

$$xy \leq \frac{1}{2}\left(\frac{\bar{y}}{\bar{x}}x^2 + \frac{\bar{x}}{\bar{y}}y^2\right). \tag{17}$$

By substituting $x = p_m, \bar{x} = p_m^{(i)}, y = r_m, \bar{y} = r_m^{(i)}$, we can equivalently express (16) as the following convex constraint.

$$E_m^{(i)}(\boldsymbol{\alpha}^{(i)}, \mathbf{p}, \mathbf{f}^{(i)}) = \theta_m(1-\alpha_m^{(i)})C_m(f_m^{(i)} + \hat{f}_m)^2 + \frac{\alpha_m^{(i)} S_m}{2}\left(\frac{r_m^{(i)}}{p_m^{(i)}}p_m^2 + \frac{p_m^{(i)}}{r_m^{(i)}}r_m^2, \right) \leq E_m^{\max}. \tag{18}$$

Consequently, we have successfully transformed the problem (11) into a convex problem to solve at the i-th iteration as follows.

$$\min_{\mathbf{p},\mathbf{r}|\mathbf{f}^{(i)},\boldsymbol{\alpha}^{(i)}} \max_{\forall m}\{\xi_m^{(i)}(\boldsymbol{\alpha}^{(i)}, \mathbf{p}, \mathbf{f}^{(i)})\}, \tag{19a}$$

$$\text{s.t. } (15), (18), (14), \tag{19b}$$

where $\xi_m^{(i)}(\boldsymbol{\alpha}^{(i)}, \mathbf{p}, \mathbf{f}^{(i)}) = E_m^{(i)}(\alpha_m, \mathbf{p}, \mathbf{f})\eta C_{\text{IE}}$. The problem is now can be solved efficiently with the well known CVX package.

B. Processing Rate Optimisation

The second subproblem solved in the alternating-based solution is the processing rate optimisation problem. In this subproblem, we solve for the optimal adjusting of the computing resource of the IoT devices and the edge server to execute the computational tasks. Given $(\mathbf{p}^{(i)}, \boldsymbol{\alpha}^{(i)})$, this subproblem finds the optimal processing rate, (i.e., $\mathbf{f}$). The optimisation problem is expressed as follows.

$$\min_{\mathbf{f}|\mathbf{p}^{(i)},\boldsymbol{\alpha}^{(i)}} \max_{\forall m}\{\xi_m(\boldsymbol{\alpha}^{(i)}, \mathbf{p}^{(i)}, \mathbf{f})\}, \tag{20a}$$

$$\text{s.t. } D_m(\boldsymbol{\alpha}^{(i)}, \mathbf{p}^{(i)}, \mathbf{f}) \leq D_m^{\max}, \forall m \tag{20b}$$

$$E_m(\boldsymbol{\alpha}^{(i)}, \mathbf{p}^{(i)}, \mathbf{f}) \leq E_m^{\max}, \forall m, \tag{20c}$$

$$(10e), (10f). \tag{20d}$$

As observed in problem (20), the energy consumption expression is quadratic in terms of $\mathbf{f}$, while the e2e delay expression is a combination of reciprocal functions involving the variables $\mathbf{f}$. Consequently, problem (20) is a convex program with respect to the variables $(f_m, f_m^0, \forall m)$, rendering it solvable using the CVX package.

C. Task Offloading Decisions Optimisation

Lastly, the optimisation of task offloading decisions addressed. Given $\mathbf{p}^{(i)}, \mathbf{f}^{(i)}$, this subproblem aims to identify the optimal task offloading decisions, represented by $\boldsymbol{\alpha}$. The optimisation problem can be expressed as follows.

$$\min_{\boldsymbol{\alpha}|\mathbf{p}^{(i)},\mathbf{f}^{(i)}} \max_{\forall m}\{\xi_m(\boldsymbol{\alpha}, \mathbf{p}^{(i)}, \mathbf{f}^{(i)})\}, \tag{21a}$$

$$\text{s.t. } D_m(\boldsymbol{\alpha}, \mathbf{p}^{(i)}, \mathbf{f}^{(i)}) \leq D_m^{\max}, \forall m. \tag{21b}$$

$$E_m(\boldsymbol{\alpha}^{(i)}, \mathbf{p}^{(i)}, \mathbf{f}) \leq E_m^{\max}, \forall m, \tag{21c}$$

$$\sum_{m=1}^{M} \alpha_m f_m^{0(i)} \leq F_{\max}, \tag{21d}$$

$$(10f). \tag{21e}$$

This problem poses a mixed-integer (binary) programming challenge, known for its computational complexity. Fortunately, the MOSEK solver integrated into CVX is adept at handling such problems efficiently. Hence, we can determine optimal task offloading decisions by solving (21) with the given $(\mathbf{p}^{(i)}, \mathbf{f}^{(i)})$ parameters at the i-th iteration.

D. Proposed SCA-based Algorithm

Building upon the aforementioned progress, we introduce an alternating-based optimisation algorithm designed to tackle (10), as outlined in Algorithm 1. The algorithm commences with an initialisation step, during which initial feasible points are derived using the formulations of the subproblems (19), (20), and (21). The sequence of the solving procedure is outlined as follows. First, the algorithm computes the optimal transmit power given the current values of the processing rate and offloading variables. Next, the processing rate optimisation is performed using the new transmit power solutions and the current offloading decisions. Finally, the algorithm determines the offloading decisions before commencing the next iteration.

Algorithm 1 : Proposed SCA-based algorithm for solving (10).

1: **Initialisation**: Set $i = 1$, maximum number of iteration, $I_{\max}$; generate the initial feasible points, and choose the initial parameters for (10).
2: **repeat**
3: Solve (19) with the given $\mathbf{f}^{(i)}, \boldsymbol{\alpha}^{(i)}$ to find next solutions of the transmission power;
4: Solve (20) with the given $\mathbf{p}^{(i)}, \boldsymbol{\alpha}^{(i)}$ to find next solutions of the processing rate variables;
5: Solve (21) with the given $\mathbf{p}^{(i)}, \mathbf{f}^{(i)}$ to find next solutions of the offloading decision variables;
6: **until** (*meet convergence* or $i > I_{\max}$)
7: **Solution**: optimal solutions of $\{\mathbf{p}^*, \mathbf{f}^*, \boldsymbol{\alpha}^*\}$.

Discussions of feasible points initialisation and the algorithm's complexity: It is important to note that the initialisation of feasible points plays a crucial role in the optimisation process. To achieve this, we set the transmission power and processing rate equally for all UEs, i.e., $p_m = P_{\min}, \forall m$, $f_m = F_m^{\min}, \forall m$, $f_m^0 = F_{\max}/M$, and assign the offloading decision of the first half of the UEs to 1. To ensure that all constraints in (10) are satisfied at the first iteration, we implement a helper function to verify that none of the constraints are violated before starting the optimisation process.

Regarding the complexity of Algorithm 1, the major computational load arises from solving the power control problem in (19). As presented in (19), this convex program involves $2M$ scalar variables and $3M$ constraints, resulting in a per-iteration complexity of $\mathcal{O}\left(\sqrt{3M}(2M)^2\right)$ [34]. Consequently, considering the number of iterations I, the worst-case complexity of the proposed Algorithm 1 is $\mathcal{O}\left(I\sqrt{3M}(2M)^2\right)$.

IV. Proposed DRL-based Solution for Adaptive Configuration in DT-enabled Edge Networks

In this section, we proposed a DRL-based solution for solving the problem (10) with the concept of DT technology. We assume that DT system fully replicates the physical systems, obtains necessary information to perform adaptive configuration on the transmit power, the processing rate, and the task offloading decisions.

A. Reinforcement Learning Representation

To solve the problem (10) with DRL-based solution, we must represent the problem as a Markov decisions process, consisting of state space ($\mathcal{S}$), action space ($\mathcal{A}$), and the reward function($\mathcal{R}$).

1) State space: At the time slot t, the state $\mathcal{S}_t$ captures all necessary environment information including system parameters, required CPU cycles of the tasks (C_m), and channel conditions $\mathbf{h}(t)$. Base on these information of the environment and the current selected action of the agent, we calculate the current value of the emission amount at the time slot t to present the state $\xi_m(t)$.

2) Action space: The DT agent make decisions to configure the system by optimising the transmission power, the processing rate, and the offloading decisions. Thus, the action at the time slot t is expressed as $a(t) = \{\mathbf{p}_t, \mathbf{f}_t, \boldsymbol{\alpha}_t\}$.

3) Reward function: The reward function is designed to reflex the objective function of the formulated problem, which aims at minimising the worst-case of CO_2 emission amount for task processing of the IoT devices. To do this, we design the reward function for the proposed DRL-based solution as follows.

$$r_t = -(\lambda_1 \xi_t + \lambda_2 \rho_m^D + \lambda_3 \rho_m^E + \lambda_4 \rho_m^F + \lambda_5 \rho_m^R), \qquad (22)$$

where $\xi_t = \max\{\xi_m\}_{\forall m}$ at timeslot t; λ_1, λ_2, λ_3, λ_4, λ_5 are the weight factors to balance the objectives; ρ_m^D, ρ_m^E, ρ_m^E, and ρ_m^R are the penalty for violating constraints of delay tolerance (21b), energy budget (21c), computing capacity of edge server (10e), and minimum transmission rate (10d), respectively.

B. Proposed DDPG-based Algorithm for DTCAEC

Based on the above transformation and the characteristic of the addressed optimisation, we propose a deep deterministic policy gradient (DDPG) based algorithm as presented in Algorithm 2 to solve the problem. Detailed implementation of this algorithm is provided in the following subsection.

Algorithm 2 : Proposed DDPG-based algorithm for solving (10) in DTCAEC.

Initialise actor $\mu(s|\theta^\mu)$ and critic $Q(s,a|\theta^Q)$ networks with weights θ^μ, θ^Q, respectively.
Initialise target networks μ' and Q' with weights $\theta^{\mu'}$, $\theta^{Q'}$, respectively.
Initialise replay buffer $\mathcal{B}$ and exploration noise process $\mathcal{N}$ using the Ornstein-Uhlenbeck (OU) process.
for episode $= 1$ to K **do**
  The agent receives initial state s_1 from the environment;
  for step $t = 1$ to T **do**
    Policy Execution: Get action $a_t = \mu(s_t|\theta^\mu) + \mathcal{N}_t$;
    Execute a_t, observe reward r_t and next state s_{t+1};
    Store transition (s_t, a_t, r_t, s_{t+1}) in $\mathcal{B}$;
    Experience Replay: Sample minibatch $\{(s_i, a_i, r_i, s_{i+1})\}_{i=1}^{N_b}$ from $\mathcal{B}$
    Compute target:
    $$y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1}|\theta^{\mu'})|\theta^{Q'})$$
    Update the critic network by minimising loss:
    $$L = \frac{1}{N_b}\sum_i \left(y_i - Q(s_i, a_i|\theta^Q)\right)^2$$
    Update the actor network by using policy gradient:
    $$\nabla_{\theta^\mu} J \approx \frac{1}{N_b}\sum_i \nabla_a Q(s,a|\theta^Q)|_{a=\mu(s_i)} \nabla_{\theta^\mu}\mu(s|\theta^\mu)|_{s_i}$$
    Target Network Update:
    $$\theta^{Q'} \leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'}, \quad \theta^{\mu'} \leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu'}$$
    Update state: $s_t \leftarrow s_{t+1}$
  end for
end for

Discussions of complexity and settings of penalty parameters: The complexity of the proposed DDPG algorithm primarily stems from the forward and backward passes of the actor and critic networks, and replay buffer operations. At each training step, the critic network updates its parameters by minimising the temporal difference (TD) error, while the actor network is updated using the policy gradient. These operations have complexities of $\mathcal{O}(N_a)$ and $\mathcal{O}(N_c)$, where N_a and N_c are the numbers of parameters in the actor and critic networks, respectively. Sampling a minibatch of size N_b from the replay buffer adds $\mathcal{O}(N_b)$. Soft updates of the target networks contribute an additional cost of $\mathcal{O}(N_a + N_c)$. The total per-step complexity is approximately $\mathcal{O}\big(N_b(N_a + N_c)\big)$. For T time steps per episode and K episodes, the overall complexity becomes $\mathcal{O}\big(TKN_b(N_a + N_c)\big)$.

The choice of penalty parameters in the reward function, denoted as $\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5$, is crucial for balancing competing

objectives, such as CO_2 emissions, delay, energy consumption, transmission rate, and resource usage. Each parameter reflects the relative importance of its corresponding term. Proper tuning ensures the agent prioritises objectives effectively while respecting system constraints, such as $D^{\max_m}, R^{\min_m}$, and $F^{\max}$. A well-tuned reward function encourages the agent to find optimal trade-offs across objectives without violating resource limits.

C. Implementation of the DDPG-based Algorithm 2

The implementation of DDPG-based solution is constructed based on the actor-critic framework. The workflow involves interactions between multiple components, including the environment, actor network, critic network, relay buffer, and their respective target networks, as illustrated in Fig. 2.

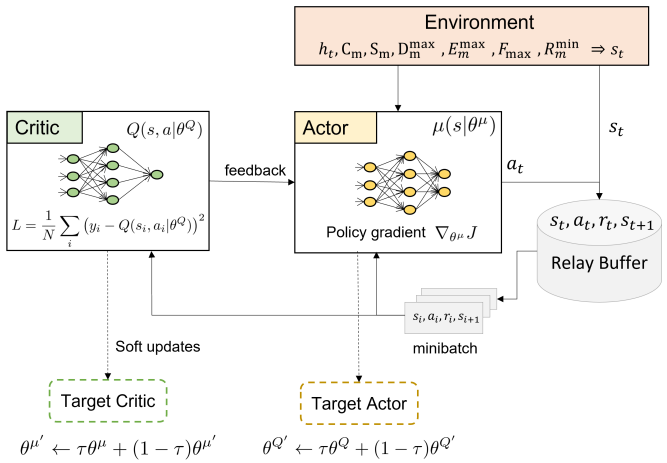

Fig. 2. An illustration of DDPG-based solution for solving the problem (10).

The environment encapsulates the system dynamics, including parameters such as $\mathbf{h}_t$ (channel state), C_m (computation requirements), S_m (data size), $D_m^{\max}$ (maximum delay), $E_m^{\max}$ (maximum energy), $F_m^{\max}$ (maximum CPU frequency), and $R_m^{\min}$ (minimum rate). At each timestep t, the agent interacts with the environment, receiving the current state s_t. Based on this state, the actor network generates an action a_t, which is executed in the environment to transition to the next state s_{t+1}. The environment also provides the reward r_t associated with the action a_t. We note that the offloading decisions are represented as binary variables. To ensure this, a rounding operation is applied during the action generation process.

The actor network in the DDPG algorithm is responsible for representing the policy $\mu(s|\theta^\mu)$, which maps the current state s_t to a deterministic action a_t. The primary goal of the actor network is to generate actions that maximise the long-term expected reward. To achieve this, the actor network is updated using the policy gradient approach:

$$\nabla_{\theta^\mu} J = \mathbb{E}_{s\sim p^\pi}[\nabla_a Q(s,a|\theta^Q)\nabla_{\theta^\mu}\mu(s|\theta^\mu)]. \tag{23}$$

This update ensures that the actor improves its policy by leveraging the feedback from the critic network. The feedback is a signal that indicates how the policy should change to maximise the expected reward, making the actor adapt iteratively to achieve better performance.

The critic network, on the other hand, evaluates the quality of the actions produced by the actor using the Q-function $Q(s,a|\theta^Q)$. It learns to approximate the expected return of taking a particular action a in a given state s and following the policy thereafter. The critic is trained by minimising the TD error:

$$L = \frac{1}{N_b}\sum_i (y_i - Q(s_i,a_i|\theta^Q))^2, \tag{24}$$

where:

$$y_i = r_i + \gamma Q(s_{i+1}, \mu(s_{i+1}|\theta^{\mu'})|\theta^{Q'}).$$

The critic's evaluation is critical for providing a learning signal to the actor, guiding it toward actions that yield higher rewards. By continuously refining its Q-value estimates, the critic ensures accurate feedback, which forms the basis for the actor's updates. Together, the actor and critic networks form a tightly coupled system where the critic guides the actor's learning, and the actor refines its policy to align with the critic's evaluations.

The relay buffer is a key component of the DDPG algorithm. It stores past experiences in the form of tuples (s_t, a_t, r_t, s_{t+1}). During training, a minibatch of experiences is sampled randomly from the relay buffer to compute gradients and update the parameters of the actor and critic networks. This mechanism improves training stability by decorrelating consecutive samples and ensuring efficient reuse of past data. The use of the replay buffer helps the agent learn more robustly from diverse samples, rather than relying solely on the most recent interactions. Mathematically, the experiences stored in the relay buffer are used to compute the loss functions for the critic and actor networks, which are subsequently minimised using gradient descent.

Finally, the target networks play a crucial role in stabilising the training process. They include the *Target Actor* $\mu'(s|\theta^{\mu'})$ and the *Target Critic* $Q'(s,a|\theta^{Q'})$. These networks provide stable targets for updating the primary actor and critic networks. The parameters of the target networks are updated slowly to track their respective primary networks using a soft update mechanism:

$$\theta^{\mu'} \leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu'},$$
$$\theta^{Q'} \leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'}.$$

Here, τ is the soft update parameter that determines the rate at which the target networks adjust to changes in the primary networks. By providing stable targets, the target networks mitigate the problem of oscillations during training and help ensure convergence.

V. Simulation Results and Discussions

A. Simulation Setting

For simulations, we consider a small-scale IoT network, where all IoT devices are randomly distributed in a space of 100 m x 100 m. There are totally 6 devices connecting to the BS in the network. The large-scale fading for the

wireless transmission from the m-th UE to the BS is modelled as $g_{mk} = 10^{\mathrm{PL}(\mathrm{d_{mk}})/10}$, where $\mathrm{PL}(d_{mk}) = -35.3 - 37.6 \log_{10} d_{mk}$ [32]. The single-sided noise spectral density is set to -174 dBm/Hz [32]. The task size is set in a range $[1, 4]$ MB and the delay tolerance of the task is set to 2 seconds. The maximum required CPU cycles of the task is set in the range of $[1000, 2000]$ megacycles. For the training of Algorithm 2, we set the discount factor to 0.99 and soft update parameter to 0.001. The neural networks of the actor and the critic consist of three fully connected hidden layers with $[512, 256, 128]$ neurons. Other parameters are provided in Table I. The implementation of the SCA-based solution was conducted in the MATLAB environment using the CVX package, while the DRL-based solution was developed in Python, leveraging packages such as PyTorch and Gymnasium.

For performance comparisons, we implement several benchmark schemes as follows:

- Equal processing rate (*EPR-scheme*): This scheme considers a typical edge computing system model with joint task offloading and power control optimisation; however, it does not account for adaptive computing frequency or processing rate [35], [36].
- Equal processing rate and equal power (*EPR-EP scheme*): This non-optimal scheme is designed as a conventional benchmark to demonstrate the superiority of the proposed solutions.

TABLE I
SIMULATION PARAMETERS [10], [24], [32].

Parameters	Value
Maximum transmission power of UE	$P_m^{\max} = 23$ dBm
Number of antennas of BS	$L = 8$
System bandwidth	$B = 10$ MHz
UE's processing rate range	$[0.5, 2]$ GHz
Computing capacity of edge server	$F_m^{\max} = 20$ GHz
Minimum data rate	$R^{\min} = 1$ Mbps
Maximum UE's energy consumption	$E_m^{\max} = 1$ Joule
Energy consumption parameter	$\theta_m = 10^{-27}$ Watt.s^3/cycle^3
Carbon intensity of electricity	$C_{\mathrm{IE}} = 182$ g CO_2/kWh
Energy conversion parameter	$\eta = 2.77778e^{-7}$

B. Numerical Results and Discussions

In this subsection, we present the numerical simulation results to illustrate the convergence patterns of the proposed algorithms and to highlight the superiority of the proposed solution in minimising CO_2 emissions. Additionally, we analyse the impact of various system parameters on the resulting CO_2 emissions.

1) Convergence pattern of the proposed Algorithm 1: To illustrate the convergence pattern of the proposed SCA-based algorithm, we monitor the worst-case CO_2 emissions among User Equipments (UEs) over the duration of its execution. As depicted in Figure 3, Algorithm 1 demonstrates its effectiveness in minimising CO_2 emissions, achieving a reduction of nearly 30 times in emissions after just 10 iterations. Notably,

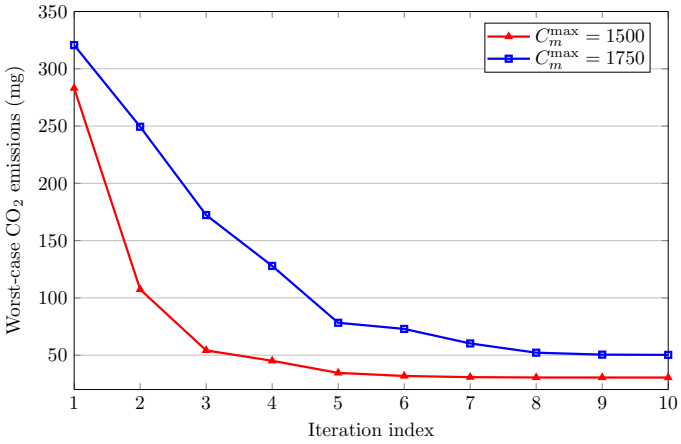

Fig. 3. Convergence pattern of Algorithm 1.

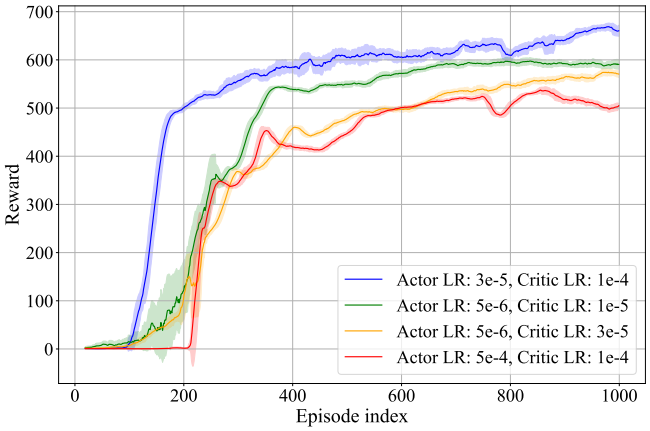

Fig. 4. Impact of learning rate on the training performance of Algorithm 2 with $C_n \sim U(1000, 1100)$.

a significant reduction in emissions occurs after the initial iteration. This phenomenon can be attributed to the algorithm's ability to initiate optimisation from points that are considerably distant from the optimal solutions. Consequently, there is ample room for improvement during the first iteration. From the fifth iteration onwards, there is a gradual decrease in CO_2 emissions until convergence is reached. This observed trend underscores the algorithm's iterative refinement process, where adjustments are made iteratively to approach the optimal solution. This convergence behaviour underscores the algorithm's potential to significantly enhance the sustainability of edge computing systems by efficiently managing resource allocation to minimise carbon emissions.

2) Impact of learning rate on the training performance of Algorithm 2: Fig. 4 illustrates the impact of varying learning rate settings on the training performance of Algorithm 2, measured in terms of the reward progression over 1000 episodes. The results show that the combination of Actor LR: 3×10^{-5}, Critic LR: 1×10^{-4} achieves the best performance, with the reward converging rapidly and reaching the highest values, maintaining stability throughout the training. The configuration with Actor LR: 5×10^{-6}, Critic LR: 1×10^{-5} follows closely, displaying a slower convergence initially but

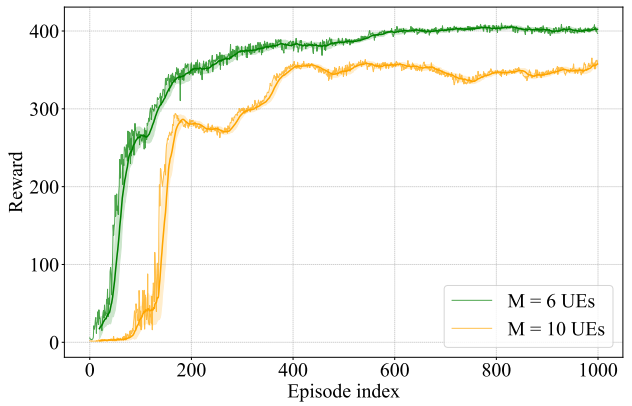

Fig. 5. Training performance of the proposed DDPG-based Algorithm 2 in the scenarios of 6 UEs and 10 UEs with $C_m \sim U(1500, 2000)$.

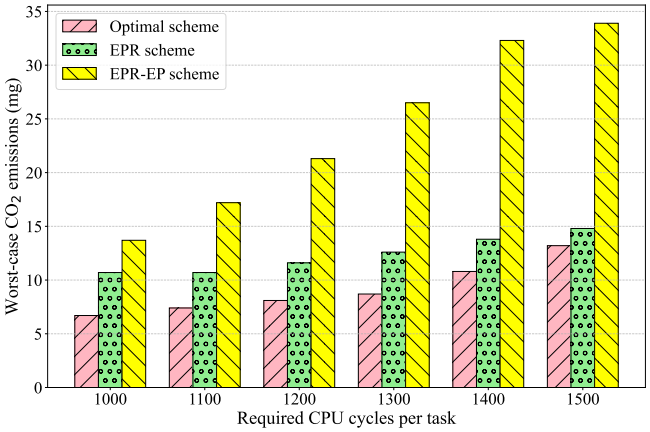

Fig. 7. Effectiveness of the proposed solution compared with non-optimal schemes.

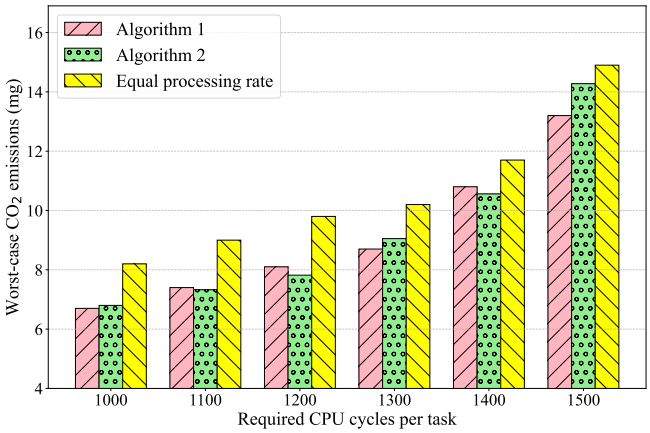

Fig. 6. Performance comparison of the proposed algorithms.

eventually achieving comparable performance. On the other hand, Actor LR: 5×10^{-6}, Critic LR: 3×10^{-5} and Actor LR: 5×10^{-4}, Critic LR: 1×10^{-4} demonstrate lower performance and less stability, as seen in their slower convergence rates and plateauing at lower reward values. This comparison highlights that carefully tuning the learning rates, particularly for the actor and critic networks, is essential for achieving both fast convergence and optimal performance.

3) Training performance of the proposed DDPG-based solution with various of number of UEs: Fig. 5 compares the training performance of the proposed DDPG-based Algorithm 2 under two scenarios: one with 6 UEs and another with 10 UEs. The reward progression shows that both scenarios converge after approximately 200 episodes. The 6 UEs scenario achieves a higher reward and stabilises around 400, while the 10 UEs scenario stabilises at a lower reward level, around 350. This performance gap can be attributed to the increased complexity of managing a higher number of UEs. The results indicate that the proposed algorithm performs effectively, but the reward decreases as the system complexity increases.

4) Performance comparison of the proposed algorithms: Fig. 6 compares the worst-case CO2 emissions for three schemes: Algorithm 1, Algorithm 2, and a non-optimal computing rate scheme. Both proposed algorithms consistently achieve better performance than the non-optimal scheme, resulting in significantly lower CO_2 emissions across all task requirements. As the complexity of the task increases, reflected by the rise in required CPU cycles, the CO_2 emission amount steadily increases for all approaches. This trend highlights the importance of developing optimisation solutions, such as the proposed algorithms, to effectively reduce CO_2 emissions. Furthermore, the results show that in most cases, the two proposed algorithms achieve approximately optimal values, with Algorithm 2 closely matching the performance of Algorithm 1.

5) Effectiveness of the proposed solutions: Fig.7 compares the worst-case CO_2 emissions (mg) for three schemes—optimal scheme, EPR scheme, and EPR-EP scheme—across varying required CPU cycles per task, ranging from 1000 to 1500. The Optimal scheme consistently achieves the lowest CO_2 emissions, demonstrating its effectiveness. The EPR scheme, which represents the equal processing rate scheme, produces higher emissions than the optimal scheme but performs better than the EPR-EP scheme. The EPR-EP scheme, representing the equal processing rate and equal power scheme, consistently yields the highest CO_2 emissions across all CPU cycle values, indicating its inefficiency compared to the other approaches. As the required CPU cycles per task increase, the CO_2 emissions for all schemes rise steadily, further highlighting the importance of the proposed optimal solution in mitigating emissions compared to the two non-optimal schemes.

6) Impacts of DT estimation: Fig. 8 highlights the critical importance of DT estimation accuracy in achieving optimal results. The data clearly shows that more precise DT estimations of processing rates lead to significantly lower CO_2 emissions. This finding demonstrates the essential role of accurate DT models and latency estimation in refining resource allocation strategies and reducing environmental impact. Furthermore, Fig. 8 illustrates the relationship between task complexity, represented by the required number of CPU cycles, and CO_2 emissions. As task complexity increases, CO_2 emissions rise accordingly, emphasising the direct influence of computational intensity on the environmental footprint. These observations

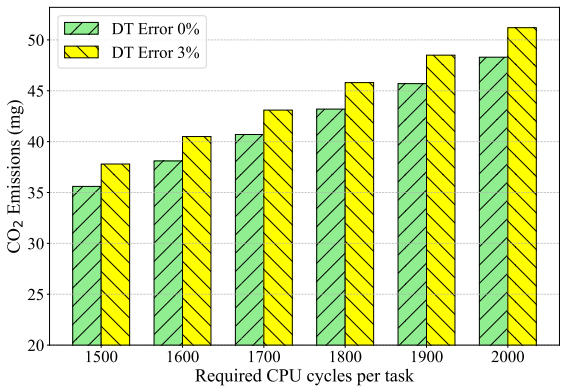

Fig. 8. Impacts of DT estimations.

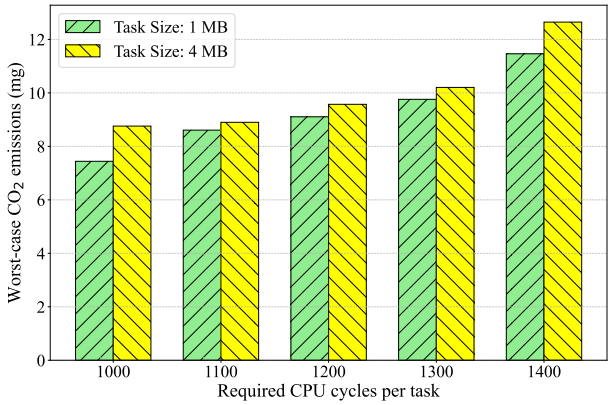

Fig. 9. Impacts of task size and task complexity.

collectively highlight the necessity of improving DT accuracy to enhance performance and sustainability in system operations.

7) Impacts of task size and task complexity: Fig. 9 illustrates the impacts of task size and task complexity on the worst-case CO_2 emissions (mg) as the required CPU cycles per task increase. Two task sizes, 1 MB and 4 MB, are compared across CPU cycle values ranging from 1000 to 1400. The results show that larger task sizes lead to higher CO_2 emissions for all CPU cycle settings. This is attributed to the increased transmission latency i.e., (3) and energy consumption i.e., (7) associated with larger task sizes, which contribute significantly to the overall emissions. Furthermore, as the required CPU cycles increase, the CO_2 emissions steadily rise for both task sizes. This demonstrates that both task size and task complexity play crucial roles in influencing CO_2 emissions, with larger tasks and higher computational demands resulting in greater environmental impact.

VI. Conclusion

In summary, this paper has examined sustainable resource management within edge computing systems, employing the DT approach. The primary objective has been to reduce CO_2 emissions in MEC-aided IoT scenarios by optimising key factors, such as IoT device transmit power, DT-estimated processing rates, and task offloading decisions. To address the complex challenge, we has proposed two solutions: the alternating optimisation algorithm based on the SCA framework and the DRL-based solution. The DRL-based approach leverages the adaptability of reinforcement learning to manage dynamic environments and optimise resource allocation decisions effectively. Through extensive simulations, the effectiveness of the proposed solutions has been validated. Both approaches demonstrated significant reductions in CO_2 emissions and optimised resource allocation. These results highlight the potential of combining DT and AI-driven methods to enhance sustainability in edge computing environments. Looking ahead, there is a promising avenue for further research, particularly in scaling these approaches to accommodate large-scale networks. Future efforts could focus on integrating advanced machine learning techniques and data-driven algorithms to meet the evolving demands of modern computing systems while continuing to minimise their environmental impact.

References

[1] D. V. Huynh, S. R. Khosravirad, V. Sharma, B. Canberk, O. A. Dobre, and T. Q. Duong, "Digital twin-enabled low-carbon sustainable edge computing for wireless networks," in *Proc. IEEE Global Commun. Conf.*, Cape Town, South Africa, Dec. 8–12, 2024.

[2] K. B. Letaief, Y. Shi, J. Lu, and J. Lu, "Edge artificial intelligence for 6G: Vision, enabling technologies, and applications," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 1, pp. 5–36, Jan. 2022.

[3] K.-C. Chen, S.-C. Lin, J.-H. Hsiao, C.-H. Liu, A. F. Molisch, and G. P. Fettweis, "Wireless networked multirobot systems in smart factories," *Proc. IEEE*, vol. 109, no. 4, pp. 468–494, Apr. 2021.

[4] Q. Luo, S. Hu, C. Li, G. Li, and W. Shi, "Resource scheduling in edge computing: A survey," *IEEE Commun. Surveys Tuts.*, vol. 23, no. 4, pp. 2131–2165, Fourthquarter 2021.

[5] L. Lin, X. Liao, H. Jin, and P. Li, "Computation offloading toward edge computing," *Proc. IEEE*, vol. 107, no. 8, pp. 1584–1607, Aug. 2019.

[6] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2322–2358, Fourthquarter 2017.

[7] T. Q. Duong, D. V. Huynh, S. R. Khosravirad, V. Sharma, O. A. Dobre, and H. Shin, "From digital twin to metaverse: The role of 6G ultra-reliable and low-latency communications with multi-tier computing," *IEEE Wireless Commun.*, vol. 30, no. 3, pp. 140–146, Jun. 2023.

[8] T. Liu, L. Tang, W. Wang, Q. Chen, and X. Zeng, "Digital twin assisted task offloading based on edge collaboration in the digital twin edge network," *IEEE Internet Things J.*, no. 2, pp. 1427–1444, Jan. 2022.

[9] B. Cao, Z. Li, X. Liu, Z. Lv, and H. He, "Mobility-aware multiobjective task offloading for vehicular edge computing in digital twin environment," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 10, pp. 3046–3055, Oct. 2023.

[10] T. Do-Duy, D. V. Huynh, O. A. Dobre, B. Canberk, and T. Q. Duong, "Digital twin-aided intelligent offloading with edge selection in mobile edge computing," *IEEE Wireless Commun. Lett*, vol. 11, no. 4, pp. 806–810, Apr. 2022.

[11] Y. Zhang, J. Hu, and G. Min, "Digital twin-driven intelligent task offloading for collaborative mobile edge computing," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 10, pp. 3034–3045, Oct. 2023.

[12] D. V. Huynh, S. R. Khosravirad, A. Masaracchia, O. A. Dobre, and T. Q. Duong, "Edge intelligence-based ultra-reliable and low-latency communications for digital twin-enabled metaverse," *IEEE Wireless Commun. Lett*, vol. 11, no. 8, pp. 1733–1737, Aug. 2022.

[13] M. Golec, S. S. Gill, F. Cuadrado, A. K. Parlikad, M. Xu, H. Wu, and S. Uhlig, "ATOM: AI-powered sustainable resource management for serverless edge computing environments," *IEEE Trans. Sustain. Comput.*, vol. 9, no. 6, pp. 817–829, Nov.-Dec. 2024.

[14] H. Liao, G. Tang, D. Guo, K. Wu, and L. Luo, "EV-assisted computing for energy cost saving at edge data centers," *IEEE Trans. Mobile Comput.*, vol. 23, no. 9, pp. 9029–9041, Sep. 2024.

[15] K. Chen, Y. Sun, S. Zheng, H. Yang, and P. Yu, "Online collaborative energy-network resource scheduling for WPT-enabled green edge computing," *IEEE Trans. on Green Commun. Netw.*, vol. 8, no. 2, pp. 601–618, Jun. 2024.

[16] Y. Li, D. V. Huynh, V.-L. Nguyen, D.-B. Ha, H.-J. Zepernick, and T. Q. Duong, "Multiagent UAV-aided URLLC mobile edge computing systems: A joint communication and computation optimization approach," *IEEE Syst. J.*, vol. 18, no. 4, pp. 1828–1838, Dec. 2024.

[17] W. Fan, Y. Su, J. Liu, S. Li, W. Huang, F. Wu, and Y. Liu, "Joint task offloading and resource allocation for vehicular edge computing based on V2I and V2V modes," *IEEE Trans. on Intell. Transp. Syst.*, vol. 24, no. 4, pp. 4277–4292, Apr. 2023.

[18] D. V. Huynh, V.-D. Nguyen, S. R. Khosravirad, K. Wang, G. K. Karagiannidis, and T. Q. Duong, "Distributed communication and computation resource management for digital twin-aided edge computing with short-packet communications," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 10, pp. 3008–3021, Aug. 2023.

[19] L. Zhao, Z. Zhao, E. Zhang, A. Hawbani, A. Al-Dubai, T. Tan, and A. Hussain, "A digital twin-assisted intelligent partial offloading approach for vehicular edge computing," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 11, pp. 3386–3400, Nov. 2023.

[20] L. Zhang, H. Wang, H. Xue, H. Zhang, Q. Liu, D. Niyato, and Z. Han, "Digital twin-assisted edge computation offloading in industrial Internet of things with NOMA," *IEEE Trans. Veh. Technol.*, vol. 72, no. 9, pp. 11 935–11 950, Sep. 2023.

[21] J. Li, S. Guo, W. Liang, J. Wang, Q. Chen, Y. Zeng, B. Ye, and X. Jia, "Digital twin-enabled service provisioning in edge computing via continual learning," *IEEE Trans. Mobile Comput.*, vol. 23, no. 6, pp. 7335–7350, Jun. 2024.

[22] Z. Yu, Y. Zhao, T. Deng, L. You, and D. Yuan, "Less carbon footprint in edge computing by joint task offloading and energy sharing," *IEEE Netw. Lett.*, vol. 5, no. 4, pp. 245–249, Dec. 2023.

[23] H. Ma, Z. Zhou, and X. Z. andXu Chen, "Toward carbon-neutral edge computing: Greening edge AI by harnessing spot and future carbon markets," *IEEE Internet of Things J.*, vol. 10, no. 18, pp. 16 637–16 649, Sep. 2023.

[24] L. Gu, W. Zhang, Z. Wang, D. Zeng, and H. Jin, "Service management and energy scheduling toward low-carbon edge computing," *IEEE Trans. on Sustainable Comput.*, vol. 8, no. 1, pp. 109–119, 2023.

[25] C.-S. Yang, C.-C. Huang-Fu, and I.-K. Fu, "Carbon-neutralized task scheduling for green computing networks," in *Proc. 2021 IEEE Glob. Commun. Conf.*, Rio de Janeiro, Brazil, Dec. 2022, pp. 4824–4829.

[26] Z. Yu, Y. Zhao, T. Deng, L. You, and D. Yuan, "Less carbon footprint in edge computing by joint task offloading and energy sharing," *IEEE Netw. Lett.*, vol. 5, no. 4, pp. 245–249, Dec. 2023.

[27] M. S. Aslanpour, A. N. Toosi, M. A. Cheema, and M. B. Chhetri, "faasHouse: Sustainable serverless edge computing through energy-aware resource scheduling," *IEEE Trans. Sustain. Comput.*, vol. 17, no. 4, pp. 9029–9041, July-Aug. 2024.

[28] G. Cui, Q. He, X. Xia, F. Chen, and Y. Yang, "Eesaver: Saving energy dynamically for green multi-access edge computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 34, no. 7, pp. 2155–2166, Jul. 2023.

[29] W. Sun, H. Zhang, R. Wang, and Y. Zhang, "Reducing offloading latency for digital twin edge networks in 6G," *IEEE Trans. Veh. Technol.*, vol. 69, no. 10, Oct. 2020.

[30] Y. Wu, K. Zhang, and Y. Zhang, "Digital twin networks: A survey," *IEEE Internet Things J.*, vol. 8, no. 18, pp. 13 789–13 804, Sep. 2021.

[31] J. C. Bezdek and R. J. Hathaway, "Some notes on alternating optimization," in *AFSS international conference on fuzzy systems*. Springer, 2002, pp. 288–300.

[32] A. A. Nasir, H. D. Tuan, H. Nguyen, M. Debbah, and H. V. Poor, "Resource allocation and beamforming design in the short blocklength regime for URLLC," *IEEE Trans. Wireless Commun.*, vol. 20, no. 2, pp. 1321–1335, Feb. 2021.

[33] H. H. M. Tam, H. D. Tuan, D. T. Ngo, T. Q. Duong, and H. V. Poor, "Joint load balancing and interference management for small-cell heterogeneous networks with limited backhaul capacity," *IEEE Trans. Commun.*, vol. 16, no. 2, pp. 872–884, Feb. 2017.

[34] A. Ben-Tal and A. Nemirovski, *Lectures on Modern Convex Optimization.* Philadelphia: MPS-SIAM Series on Optimi., SIAM, 2001.

[35] R. Dong, C. She, W. Hardjawana, Y. Li, and B. Vucetic, "Deep learning for hybrid 5G services in mobile edge computing systems: Learn from a digital twin," *IEEE Trans. Wireless Commun.*, vol. 18, no. 10, pp. 4692–4707, Oct. 2019.

[36] K. Zhang, J. Cao, and Y. Zhang, "Adaptive digital twin and multi-agent deep reinforcement learning for vehicular edge computing and networks," *IEEE Trans. Ind. Informat.*, vol. 18, no. 2, pp. 1405–1413, Feb. 2022.