Carbon-Neutralized Task Scheduling for Green Computing Networks

Chien-Sheng Yang, Chien-Chun Huang-Fu and I-Kang Fu
MediaTek Inc.

***Abstract*—Climate change due to increasing carbon emissions by human activities has been identified as one of the most critical threat to Earth. Carbon neutralization, as a key approach to reverse climate change, has triggered the development of new regulations to enforce the economic activities toward low carbon solutions. Computing networks that enable users to process computation-intensive tasks contribute huge amount of carbon emissions due to rising energy consumption. To analyze the achievable reduction of carbon emissions by a scheduling policy, we first propose a novel virtual queueing network model that captures communication and computing procedures in networks. To adapt to highly variable and unpredictable nature of renewable energy utilized by computing networks (i.e., carbon intensity of grid varies by time and location), we propose a novel carbon-intensity based scheduling policy that dynamically schedules computation tasks over clouds via the drift-plus-penalty methodology in Lyapunov optimization. Our numerical analysis using real-world data shows that the proposed policy achieves 54% reduction on the cumulative carbon emissions for AI model training tasks compared to the queue-length based policy.**

I. Introduction

Global warming caused by excessive emissions of carbon dioxide (e.g., burning fossil fuels for electricity generation) is the main driver to climate change, which has posed a significant threat to human society. To limit global warming, the most essential approach is via carbon neutralization, i.e., compensate carbon emissions by acquiring carbon offsets. Although the offsetting mechanisms for trading carbon credits (e.g., UN Carbon Offset Platform [1]) have been widely adopted globally, it has been shown that such mechanisms have limitations to effectively reduce the emissions [2]. To achieve carbon neutrality, it is important to reduce the carbon emissions in the first place rather than offset them later.

Due to recent advancements in computing networks that enable users to offload computation-intensive tasks to clouds, service demands for computing and communication resources in networks have been dramatically rising since 2010 [3]. Thus, carbon emissions due to increasing energy consumption in computing networks become a matter of concern. To reduce their carbon footprint and limit their environmental impacts, clouds have been pushed to use more renewable energy, e.g., Amazon AWS's goal of 100% renewable energy by 2025 [4].

Electricity generation is from energy sources (e.g., gas, coal, wind energy) with different levels of carbon emissions. In particular, due to the highly variable and unpredictable nature of renewable energy sources (e.g., solar energy), carbon intensity (i.e., average carbon emissions per unit of energy consumption) of electricity grid varies considerably by time and location [5], [6]. Thus, to guarantee the reduction of carbon emissions in computing networks, there is a critical need to design a task scheduling policy for networks, which accounts for temporal and spatial dimensions of energy sources.

In this paper, we consider the problem of task scheduling over computing networks with focus on the reduction of carbon emissions. More precisely, the considered computing network model is composed of an edge server and multiple clouds, in which the offloaded tasks arrive to the edge dynamically and then are dispatched to clouds accordingly. The edge server is responsible for sending data of tasks to clouds, and the energy consumption of edge server depends on which type of tasks it is sending. Each cloud is responsible for processing tasks, and the energy consumption of a cloud depends on which type of tasks it is processing. Subject to the energy consumption constraints, we assume that the edge server and each of clouds use different electricity grid, i.e., have different carbon intensity. To design an efficient scheduling policy that minimizes the carbon emissions from computing networks, we aim at exploiting the workload flexibility in both when and where the computation tasks are executed.

To analyze the carbon emissions from the network, we first propose a novel virtual queueing network model that captures the communication and computing procedures in the network. Then, in order to adapt to varied carbon intensity of electricity grids, we introduce the drift-plus-penalty methodology of Lyapunov optimization [7], whose idea is to minimize an upper bound on the drift-plus-penalty term (i.e., a linear combination of drifts and the carbon emissions with positive sign) at each time slot. Under the i.i.d assumption of the number of arriving tasks and the carbon intensity of edge and clouds, the introduced drift-plus-penalty methodology provides the guarantee on mean-rate stability of queues and achieves time-average carbon emissions arbitrarily close to optimal.

The minimization for the upper bound of drift-plus-penalty in our scheduling problem, however, is shown to be a NP-hard unbounded Knapsack problem. Through the greedy approach for minimizing the upper bound of drift-plus-penalty, we propose an efficient dynamic carbon-intensity based scheduling policy. Using the randomly generated data and the real-world data (from National Grid ESO [8]) of carbon intensity, we conduct the numerical studies for the case of AI model training tasks. We show that the proposed carbon-intensity based policy can significantly outperform the queue-length based policy in terms of cumulative carbon emissions, while ensuring the mean-rate stability of queues.

978-1-6654-3540-6/22/$31.00 © 2022 IEEE

GLOBECOM 2022 - 2022 IEEE Global Communications Conference | 978-1-6654-3540-6/22/$31.00 ©2022 IEEE | DOI: 10.1109/GLOBECOM48099.2022.10001542

Related Works: We provide a literature review that covers the works of online scheduling and carbon-aware network.

The online scheduling problem aims to dynamically schedule jobs that arrive to the network according to a stochastic process. One of the main goals is to find a throughput-optimal policy [9], i.e., a policy that stabilizes the network, whenever it can be stabilized. For instance, Max-Weight type policy [10] has shown to be throughput-optimal for wireless networks, flexible queueing networks [11] and dispersed computing networks [12]. Furthermore, Lyapunov optimization is a technique that minimizes drift-plus-penalty to ensure the network stability and the maximization of stochastic utility [7].

Carbon-aware network has been widely investigated in recent years to mitigate the global warming issue due to the escalating carbon emissions. One of key approaches for the reduction of carbon emissions is to do task scheduling by considering the temporal and spatial dimensions of energy sources [13]–[15]. Based on the information of carbon intensity, [13] formulated a static scheduling problem for the resources usage and the placement of virtual machines via mixed-integer linear programming, and proposed a multi-level approach to minimize the carbon emissions of data centers. [15] proposed a Lyapunov-based algorithm for clouds that minimizes electricity cost and poses a limit on the carbon emissions. By delaying temporally flexible compute workloads based on the forecast of next day's carbon intensity, [14] introduced a Carbon-Intelligent Compute Management to reduce carbon footprint of clouds. To distinguish from these carbon-aware approaches, with an objective to minimize carbon emissions, our proposed policy decides when and where to execute computation tasks dynamically. Furthermore, without any a-priori statistical knowledge and future predictions, the proposed policy is only based on the current state of computing network, i.e., number of arriving tasks, number of waiting tasks and real-time carbon intensity.

Notation: We denote by $[N]$ the set of $\{1, 2, \ldots, N\}$ for any positive integer N. We denote by $\mathbb{N}^0$ the set of non-negative integers, i.e., $\mathbb{N}^0 = \{0, 1, \ldots\}$.

II. System Model

We consider a computing network in which there is an edge server connecting to some clouds. Users offload their computation tasks to the edge server in an online manner, and then each computation task is executed by one of clouds. In particular, the electricity grids of network generate carbon emissions, due to the energy consumption for providing services.

In the network, there is one edge server and N clouds. We consider M types of computation tasks, which are possibly offloaded to the system by users. We consider the system in discrete time (i.e., $t = 0, 1, \ldots$). Let $a_m(t)$ be the number of type-m tasks that arrive to the edge at time t. For each type $m \in [M]$, we denote by p^{e}_m the energy consumption incurred by the edge server for sending a type-m task to one of clouds. We denote by $p^{\mathrm{c}}_{m,n}$ the energy consumption incurred by cloud n for processing an offloaded type-m task. At each time slot, we assume that the edge server has constant energy constraint P^{e} for communicating data to clouds, and each cloud $n \in [N]$ has constant energy constraint P^{c}_n for processing tasks.

Carbon intensity, defined as the amount of carbon emissions per unit of energy consumption (e.g., gCO2 per kW·h) is used to estimate the amount of carbon emissions incurred by the computing and communication procedures in the network. We assume that the edge server and each cloud utilize different energy sources including non-renewables (e.g., fossil) and renewables (e.g., wind), which have variation in carbon intensity. Specifically, we denote by $C^{\mathrm{e}}(t)$ the carbon intensity of grid utilized by the edge server at time t, and denote by $C^{\mathrm{c}}_n(t)$ the carbon intensity of grid utilized by cloud $n \in [N]$ at time t.

A. Problem Statement

In the task scheduling problem of computing network, a scheduling policy determines the followings: 1) when each task is sent to one of clouds, 2) the destination of each task, and 3) when each task is processed. Concretely, we define the following terms to characterize a scheduling policy. We denote by $d_{m,n}(t)$ the number of type-m tasks that are sent to cloud n at time t, and denote by $w_{m,n}(t)$ the number of type-m tasks that are processed by cloud n at time t. That is, at time t, a scheduling policy determines an action which is composed of $\{d_{m,n}(t)\}_{\forall m\in[M],\ \forall n\in[N]}$ and $\{w_{m,n}(t)\}_{\forall m\in[M],\ \forall n\in[N]}$. Let $P^{\mathrm{e}}_{\mathrm{total}}(t)$ be the total energy consumption by the edge server and $P^{\mathrm{c}}_{n,\mathrm{total}}(t)$ be the total energy consumption by cloud n, which can be written as follows:

$$P^{\mathrm{e}}_{\mathrm{total}}(t) = \sum_{m=1}^{M}\sum_{n=1}^{N} d_{m,n}(t) p^{\mathrm{e}}_m; \tag{1}$$

$$P^{\mathrm{c}}_{n,\mathrm{total}}(t) = \sum_{m=1}^{M} w_{m,n}(t) p^{\mathrm{c}}_{m,n},\ \forall n \in [N]. \tag{2}$$

An action is *feasible* if the constraints on energy consumption are satisfied[1], i.e.,

$$P^{\mathrm{e}}_{\mathrm{total}}(t) \le P^{\mathrm{e}}; \tag{3}$$

$$P^{\mathrm{c}}_{n,\mathrm{total}}(t) \le P^{\mathrm{c}}_n,\ \forall n \in [N]. \tag{4}$$

We denote by $C(t)$ the carbon emissions of the computing network at time t. Based on carbon intensity $C^{\mathrm{e}}(t)$ and $C^{\mathrm{c}}_n(t)$ at time t, $C(t)$ can be written as follows:

$$C(t) = C^{\mathrm{e}}(t) \cdot P^{\mathrm{e}}_{\mathrm{total}}(t) + \sum_{n=1}^{N} C^{\mathrm{c}}_n(t) \cdot P^{\mathrm{c}}_{n,\mathrm{total}}(t). \tag{5}$$

Definition 1 (Time-Average Carbon Emissions)**.** Given carbon emissions $C(t)$ at each time t, the time-average carbon emissions, denoted by $\bar{C}$, is defined as follows:

$$\bar{C} = \limsup_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}\left[C(t)\right]. \tag{6}$$

Based on the above system model, our main goal is to design a scheduling policy that chooses a feasible action on both when and where the computation tasks are executed at each time to minimize time-average carbon emissions $\bar{C}$.

[1]We assume that the scheduled tasks will be successfully communicated (processed) if the energy constraint of edge server (cloud) is satisfied.

III. Virtual Queueing Network Model

To analyze the resulting carbon emissions using a scheduling policy, we model a virtual queueing network that encodes the state of the computing network. Then, we introduce an optimization problem that ensures the minimization of carbon emissions and the mean-rate stability of queues.

As shown in Fig. 1, the proposed virtual queueing network consists of two kinds of queues, *edge queue* and *cloud queue*, which are modeled in the following manner:

- **Edge Queue**: We maintain one virtual queue called edge queue m for type-m tasks located in the edge server.
- **Cloud Queue**: We maintain one virtual queue called cloud queue (m,n) for type-m tasks processed by cloud n.

We describe the dynamics of the virtual queues in the network. The type-m tasks are sent to edge queue m when arriving to the edge server. The tasks in edge queue m are sent to cloud queue (m,n) if the type-m tasks are scheduled on cloud n for processing. We denote by $Q_m^{\mathrm{e}}(t)$ the length of edge queue m and denote by $Q_{m,n}^{\mathrm{c}}(t)$ the length of cloud queue (m,n) at time t. We state the dynamics of the proposed queueing network as follows. For $\forall m \in [M]$, we have

$$Q_m^{\mathrm{e}}(t+1) = \max(Q_m^{\mathrm{e}}(t) - \sum_{n=1}^{N} d_{m,n}(t), 0) + a_m(t). \quad (7)$$

For $\forall m \in [M], \forall n \in [N]$, we have

$$Q_{m,n}^{\mathrm{c}}(t+1) = \max(Q_{m,n}^{\mathrm{c}}(t) - w_{m,n}(t), 0) + d_{m,n}(t). \quad (8)$$

Now, we introduce an optimization problem called *carbon-aware queueing network planning problem (CQNPP)* that minimizes time-average carbon emissions $\bar{C}$ and stabilize all the queues in the virtual queueing network:

Carbon-Aware Queueing Network Planning Problem

$$\min \quad \bar{C} \quad (9)$$

$$s.t. \lim_{T\to\infty} \frac{\mathbb{E}[Q_m^{\mathrm{e}}(T)]}{T} = 0, \ \forall m \in [M]; \quad (10)$$

$$\lim_{T\to\infty} \frac{\mathbb{E}[Q_{m,n}^{\mathrm{c}}(T)]}{T} = 0, \ \forall m \in [M], \ \forall n \in [N]; \quad (11)$$

$$P_{\text{total}}^{\mathrm{e}}(t) \le P^{\mathrm{e}}; \quad (12)$$

$$P_{n,\text{total}}^{\mathrm{c}}(t) \le P_n^{\mathrm{c}}, \ \forall n \in [N]; \quad (13)$$

$$d_{m,n}(t), w_{m,n}(t) \in \mathbb{N}^0, \ \forall m \in [M], \ \forall n \in [N]. \quad (14)$$

In CQNPP, (10) and (11) indicate that we make each queue mean-rate stable; (12), (13) and (14) define the space of feasible actions. The proposed CQNPP is a sequential decision-making problem, which is in general challenging to solve.

IV. Carbon-Intensity Based Scheduling Policy

In this section, we introduce the drift-plus-penalty methodology in Lyapunov optimization [7] to effectively minimize the carbon emissions and make queues mean-rate stable. Then, we design an efficient carbon-intensity based scheduling policy which dynamically decides "where" and "when" tasks are processed based on the current state of network, without any a-priori statistical knowledge and future predictions.

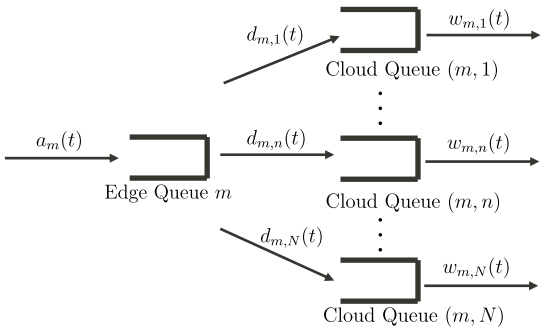

Fig. 1: The illustration of proposed queueing network. For each task type $m \in [M]$, we maintain an edge queue m and cloud queues (m,n), $\forall n \in [N]$. At time t, $a_m(t)$ number of type-m tasks arrive to edge queue m. Based on a scheduling policy, $d_{m,n}(t)$ number of type-m tasks in edge queue m arrive to cloud queue (m,n), and $w_{m,n}(t)$ number of type-m tasks depart from cloud queue (m,n).

A. Drift-Plus-Penalty Methodology

We now introduce the drift-plus-penalty methodology for the proposed CQNPP. As a measure of congestion in virtual queues, Lyapunov function $L(t)$ is defined as follows:

$$L(t) = \frac{1}{2}\left(\sum_{m=1}^{M} Q_m^{\mathrm{e}}(t)^2 + \sum_{m=1}^{M}\sum_{n=1}^{N} Q_{m,n}^{\mathrm{c}}(t)^2\right). \quad (15)$$

Then, we define the drift of Lyapunov function $L(t)$ as follows

$$\Delta(t) = L(t+1) - L(t). \quad (16)$$

To stabilize all the queues and minimize the carbon emissions, the key idea is to minimize the drift-plus-penalty, which is a weighted sum of drift and scaled penalty. Consider a non-negative number V, we formally define the drift-plus-penalty as $\Delta(t) + VC(t)$, where the penalty term at time t is carbon emissions $C(t)$. Rather than directly minimize $\Delta(t) + VC(t)$ every slot t, we minimize an upper bound on this drift-plus-penalty expression. The following lemma provides an upper bound on the drift-plus-penalty.

Lemma 1 (Drift Bound)**.** *Suppose $a_m(t)$ is upper-bounded for all m and all t. For any scheduling policy, drift-plus-penalty $\Delta(t) + VC(t)$ can be upper-bounded as follows*

$$\begin{aligned}\Delta(t) + VC(t) \le B &+ \sum_{m=1}^{M} Q_m^{e}(t) a_m(t) \\ &+ \sum_{m=1}^{M}\sum_{n=1}^{N} \left(VC^{e}(t)p_m^{e} + Q_{m,n}^{c}(t) - Q_m^{e}(t)\right) d_{m,n}(t) \\ &+ \sum_{m=1}^{M}\sum_{n=1}^{N} \left(VC_n^{c}(t)p_{m,n}^{c} - Q_{m,n}^{c}(t)\right) w_{m,n}(t) \quad (17)\end{aligned}$$

where B is a constant such that

$$\begin{aligned}&\sum_{m=1}^{M} a_m(t)^2 + \sum_{m=1}^{M}\left(\sum_{n=1}^{N} d_{m,n}(t)\right)^2 \\ &+ \sum_{m=1}^{M}\sum_{n=1}^{N} d_{m,n}(t)^2 + \sum_{m=1}^{M}\sum_{n=1}^{N} w_{m,n}(t)^2 \le 2B, \ \forall t. \quad (18)\end{aligned}$$

The proof of Lemma 1 is provided in Appendix A.

Remark 1. *We note that constant B defined in* (18) *must exist since $a_m(t)$ is assumed to be upper-bounded and $d_{m,n}(t)$ and $w_{m,n}(t)$ are subject to the constraints defined in* (12) *and* (13)*.*

At each time t, given number of arriving tasks $a_m(t)$, virtual queue-lengths $Q^{\mathrm{e}}_m(t)$, $Q^{\mathrm{c}}_{m,n}(t)$ and carbon intensity $C^{\mathrm{e}}(t)$, $C^{\mathrm{c}}_n(t)$, a policy denoted by η aims at choosing a feasible action that minimizes the upper bound defined in (17). This is equivalent to minimize

$$\sum_{m=1}^{M}\sum_{n=1}^{N}\left(VC^{\mathrm{e}}(t)p^{\mathrm{e}}_m + Q^{\mathrm{c}}_{m,n}(t) - Q^{\mathrm{e}}_m(t)\right)d_{m,n}(t) + \sum_{m=1}^{M}\sum_{n=1}^{N}\left(VC^{\mathrm{c}}_n(t)p^{\mathrm{c}}_{m,n} - Q^{\mathrm{c}}_{m,n}(t)\right)w_{m,n}(t) \tag{19}$$

where $d_{m,n}(t)$ and $w_{m,n}(t)$ are subject to (12), (13) and (14).

The following theorem shows that the theoretical guarantees provided by policy η when the number of arriving tasks and the carbon intensity are i.i.d over time slots.

Theorem 1. *Suppose $a_m(t)$ is upper-bounded for all m and all t. If $a_m(t), \forall m$, $C^e(t)$ and $C^c_n(t), \forall n$ are i.i.d over time slots, scheduling policy η with a non-negative number V that minimizes* (19) *provides the following guarantees:*

- ***Performance Guarantee.*** *The achieved time-average carbon missions $\bar{C}^{\eta}$ satisfies*

$$\bar{C}^{\eta} \leq \bar{C}^{opt} + \frac{B}{V} \tag{20}$$

  where B is the constant such that (18) *is satisfied for all t, and $\bar{C}^{opt}$ is the infimum time-average carbon emissions achievable by any policy.*
- ***Stability Guarantee.*** *All queues are mean-rate stable.*

The proof of Theorem 1 follows the similar arguments in [7], and we thus omit it due to the page limit.

Remark 2. *Theorem 1 shows that policy η achieves the time-average carbon emissions which deviates from the optimal value by no more than $\frac{B}{V}$.*

Now, we show that the minimization of (19) can not be solved efficiently. Since the edge server and the clouds have independent constraints (12) and (13), minimizing (19) can be decoupled into some independent optimization problems. For the edge server, we have the problem defined as

$$\min \sum_{m=1}^{M}\sum_{n=1}^{N} b_{m,n}(t)d_{m,n}(t); \tag{21}$$

$$s.t. \sum_{m=1}^{M}\sum_{n=1}^{N} d_{m,n}(t)p^{\mathrm{e}}_m \leq P^{\mathrm{e}}; \tag{22}$$

$$d_{m,n}(t) \in \mathbb{N}^0,\ \forall m \in [M],\ \forall n \in [N]; \tag{23}$$

and for each cloud n, we have the problem defined as

$$\min \sum_{m=1}^{M} c_{m,n}(t)w_{m,n}(t); \tag{24}$$

$$s.t. \sum_{m=1}^{M} w_{m,n}(t)p^{\mathrm{c}}_{m,n} \leq P^{\mathrm{c}}_n; \tag{25}$$

$$w_{m,n}(t) \in \mathbb{N}^0,\ \forall m \in [M],\ \forall n \in [N] \tag{26}$$

where $b_{m,n}(t) = VC^{\mathrm{e}}(t)p^{\mathrm{e}}_m + Q^{\mathrm{c}}_{m,n}(t) - Q^{\mathrm{e}}_m(t)$ and $c_{m,n}(t) = VC^{\mathrm{c}}_n(t)p^{\mathrm{c}}_{m,n} - Q^{\mathrm{c}}_{m,n}(t)$ are fixed numbers after knowing all the queue-lengths and carbon intensity at time t.

Since the problem defined in (21) to (23) aims at minimizing an objective function, the optimal solution requires that $d_{m,n}(t) = 0$ if $b_{m,n}(t) > 0$. After dropping $b_{m,n}(t)d_{m,n}(t)$'s with $b_{m,n}(t) > 0$ by setting $d_{m,n}(t) = 0$, the problem in (21) to (23) with remaining variables is an unbounded Knapsack problem which has been shown NP-hard [16]. The similar arguments also hold for the problem defined in (24) to (26).

B. Description of the Proposed Policy

We propose a carbon-intensity based scheduling policy (see Algorithm 1), whose idea is to greedily schedule tasks starting from the tasks with the most negative values contributed to (19) per energy unit. The proposed policy at each time t is dominated by the sorting procedures, which can be done efficiently (with the complexity almost linear in MN). Now, we provide more details of the proposed policy:

- **Edge Server:** For each m, we find $n_1(m)$ such that $VC^{\mathrm{e}}(t)p^{\mathrm{e}}_m + Q^{\mathrm{c}}_{m,n_1(m)}(t) - Q^{\mathrm{e}}_m(t)$ is the smallest among all n (equivalent to find $n_1(m)$ such that $Q^{\mathrm{c}}_{m,n_1(m)}(t)$ is the smallest among all n). Then, we sort the task types in increasing order of ratio $\frac{VC^{\mathrm{e}}(t)p^{\mathrm{e}}_m + Q^{\mathrm{c}}_{m,n_1(m)}(t) - Q^{\mathrm{e}}_m(t)}{p^{\mathrm{e}}_m}$ (equivalent to sort the task types in increasing order of ratio $\frac{Q^{\mathrm{c}}_{m,n_1(m)}(t) - Q^{\mathrm{e}}_m(t)}{p^{\mathrm{e}}_m}$). Subject to energy constraint P^{e}, the edge server sends as many as possible of type-m tasks to cloud $n_1(m)$ with the smallest value of $\frac{Q^{\mathrm{c}}_{m,n_1(m)}(t) - Q^{\mathrm{e}}_m(t)}{p^{\mathrm{e}}_m}$ while $VC^{\mathrm{e}}(t)p^{\mathrm{e}}_m + Q^{\mathrm{c}}_{m,n_1(m)}(t) - Q^{\mathrm{e}}_m(t)$ is negative.
- **Cloud:** For each cloud n, we sort the task types in increasing order of ratio $\frac{VC^{\mathrm{c}}_n(t)p^{\mathrm{c}}_{m,n} - Q^{\mathrm{c}}_{m,n}(t)}{p^{\mathrm{c}}_{m,n}}$ (equivalent to sort the task types in decreasing order of ratio $\frac{Q^{\mathrm{c}}_{m,n}(t)}{p^{\mathrm{c}}_{m,n}}$). Subject to energy constraint P^{c}_n, the cloud n processes as many as possible of type-m tasks with the largest value of $\frac{Q^{\mathrm{c}}_{m,n}(t)}{p^{\mathrm{c}}_{m,n}}$ while the value of $VC^{\mathrm{c}}_n(t)p^{\mathrm{c}}_{m,n} - Q^{\mathrm{c}}_{m,n}(t)$ is negative.

V. Numerical Analysis

In this section, we demonstrate the impact of the proposed carbon-intensity based scheduling policy by simulation studies. We evaluate the effectiveness of the proposed policy in terms of the cumulative carbon emissions. We consider a network composed of an edge server and 5 clouds. The edge server has energy constraint $P^{\mathrm{e}} = 4000$ kW·h, and each cloud n has energy constraint $P^{\mathrm{c}}_n = 30000$ kW·h. We consider $M = 5$ types of AI model training tasks on ImageNet [17],

Algorithm 1: Carbon-Intensity Based Policy

Input: V, M, N, P^{e}, P_n^{c}, p_m^{e}, $p_{m,n}^{\mathrm{c}}$;
Initialization: $d_{m,n}(t)=0$, $w_{m,n}(t)=0$;
for $t \leftarrow 0,1,\ldots$ **do**
  Observe $C^{\mathrm{e}}(t)$, $C_n^{\mathrm{c}}(t)$ and $a_m(t)$;
  $n_1(m) \leftarrow \mathrm{argmin}_{n\in[N]} Q_{m,n}^{\mathrm{c}}(t)$;
  Sort: $\frac{Q_{1,n_1(1)}^{\mathrm{c}}(t)-Q_1^{\mathrm{e}}(t)}{p_1^{\mathrm{e}}} \leq \cdots \leq \frac{Q_{M,n_1(M)}^{\mathrm{c}}(t)-Q_M^{\mathrm{e}}(t)}{p_M^{\mathrm{e}}}$;
  $P \leftarrow P^{\mathrm{e}}$;
  for $m \leftarrow 1$ **to** M **do**
    if $\lfloor \frac{P}{p_m^e} \rfloor > 0$ **then**
      if $VC^e(t)p_m^e + Q_{m,n_1(m)}^c(t) - Q_m^e(t) < 0$ **then**
        $d_{m,n_1(m)}(t) \leftarrow \min(Q_m^{\mathrm{e}}(t), \lfloor \frac{P}{p_m^e} \rfloor)$;
        $P \leftarrow P - \lfloor \frac{P}{p_m^e} \rfloor p_m^{\mathrm{e}}$;
      else
        break;
      end
    end
  end
  for $n \leftarrow 1$ **to** N **do**
    $P \leftarrow P_n^{\mathrm{c}}$;
    Sort: $\frac{Q_{1,n}^{\mathrm{c}}(t)}{p_{1,n}^{\mathrm{c}}} \geq \cdots \geq \frac{Q_{M,n}^{\mathrm{c}}(t)}{p_{M,n}^{\mathrm{c}}}$;
    for $m \leftarrow 1$ **to** M **do**
      if $\lfloor \frac{P}{p_{m,n}^c} \rfloor > 0$ **then**
        if $VC_n^c(t)p_{m,n}^c - Q_{m,n}^c(t) < 0$ **then**
          $w_{m,n}(t) \leftarrow \min(Q_{m,n}^{\mathrm{c}}(t), \lfloor \frac{P}{p_{m,n}^c} \rfloor)$;
          $P \leftarrow P - w_{m,n}(t)p_{m,n}^{\mathrm{c}}$;
        else
          break;
        end
      end
    end
  end
  Update $Q_m^{\mathrm{e}}(t+1)$ and $Q_{m,n}^{\mathrm{c}}(t+1)$ according to (7) and (8)
end

whose computation and communication consumption are summarized in Table I.[2] For each m, $a_m(t)$ is randomly chosen from $\{0,1,\ldots,400\}$ at each time t.

We compare the proposed policy with a queue-length based policy that makes decisions based on queue lengths: At each time t, the edge server sends as many as possible of the tasks that are located in the longest edge queues to the shortest cloud queues, and each cloud processes as many as possible of tasks located in its longest cloud queues. Then, we consider two scenarios for carbon intensity:

1) **Random:** At each time t, each of carbon intensity $C^{\mathrm{e}}(t)$ and $C_n^{\mathrm{c}}(t)$ is randomly chosen from $\{0,1,\ldots,700\}$.
2) **Real World:** National Grid ESO [8] provides the regional carbon intensity data in the UK (per 30 mins), where 6 regions' data are used to represent the carbon intensity of the edge server and 5 clouds.

Fig. 2 and Fig. 3 provide the normalized cumulative carbon-emissions comparison with the queue-length based policy.[3] Fig. 4 provides the comparison of average length of edge queue $m=1$ under the scenario of random carbon intensity. Then, we conclude the followings:

- For the random case, the proposed policy with $V=0.05$ reduces the cumulative carbon emissions by 63%, and also ensures the mean-rate stability of queues.
- For the real-world carbon intensity data, the proposed policy with $V=0.05$ reduces the cumulative carbon emissions by 54%, which demonstrates the effectiveness of the carbon-intensity based policy in the real-world scenarios.[4]
- Fig. 2 and Fig. 4 indicate a tradeoff between carbon emissions and queueing delay provided by the underlying dirft-plus-penalty methodology.

Type	Model	$p_{m,n}^{\mathrm{c}}$ (kW·h)	p_m^{e} (kW·h)
$m=1$	ResNet50	74	3.45
$m=2$	InceptionV3	97	3.45
$m=3$	DenseNet121	54	3.45
$m=4$	SqueezeNet	16	3.45
$m=5$	MobileNetV2	5.8	3.45

TABLE I: Summary for the energy consumption of AI training tasks. It is assumed that clouds are homogeneous, i.e., $p_{m,1}^{\mathrm{c}} = \cdots = p_{m,5}^{\mathrm{c}}$.

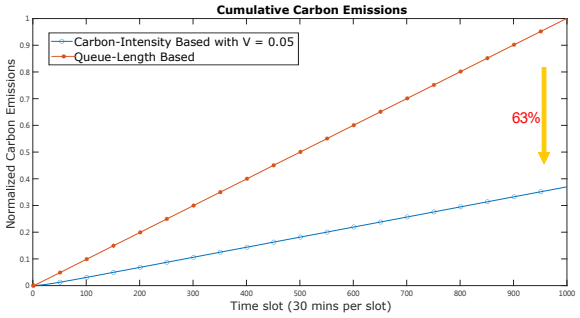

Fig. 2: Numerical evaluations for cumulative carbon emissions (normalized) with the random carbon intensity.

VI. Conclusion

In this paper, we proposed a online carbon-intensity based scheduling policy for computing networks, which utilizes the temporal and spatial information of carbon intensity to effectively reduce carbon footprint of computing and communication procedures in the networks. Moreover, the leveraged drift-plus-penalty methodology provides the tradeoff between the reduction of carbon emissions and queueing delay. The numerical analysis in our paper demonstrates that the proposed scheduling policy can effectively reduce the overall carbon emissions by 54% for AI model training tasks in the scenario of real-world carbon intensity. It is critical to take the carbon-related information into account when designing the communication and computation procedures of next-generation network in order to achieve the objective of carbon neutrality.

[2]The estimation of P_n^{c} is based on the annual energy consumption of Google [18]. The estimation of P^{e} is based on the assumptions: the bandwidth of 100 GB/s for edge and the energy efficiency of 0.023 kW·h/GB for data transmission [19]. The estimation of p_m^{e} and $p_{m,n}^{\mathrm{c}}$ are based on the size of ImageNet dataset [17] and the inference complexity of each model [20] respectively.

[3]The amount of carbon emissions will scale up in the sizes of energy consumption and energy constraint. Thus, we only focus on the normalized cumulative carbon emissions for the analysis.

[4]As indicated in [21], the total carbon emissions attributed to data centers in 2018 was 3.15×10^7 tons in the US. Thus, it is potential to reduce million tons of carbon emissions via our policy.

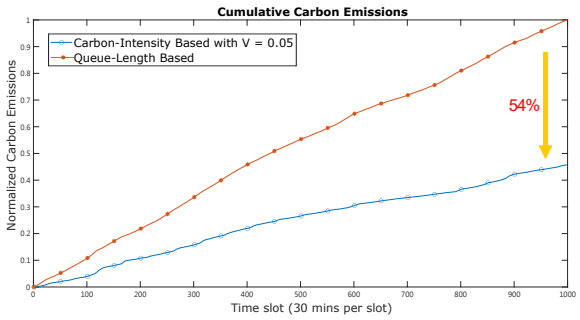

Fig. 3: Numerical evaluations for cumulative carbon emissions (normalized) with the carbon intensity from National Grid ESO [8].

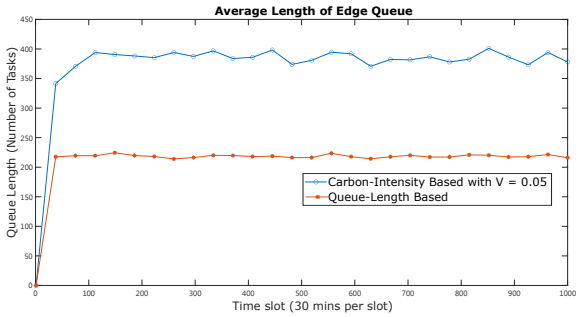

Fig. 4: Numerical evaluations for average queue length with random carbon intensity.

REFERENCES

[1] "United nations carbon offset platform." https://unfccc.int/climate-action/climate-neutral-now/united-nations-carbon-offset-platform. Accessed: 2022-03-28.

[2] "Will net-zero get us to net zero emissions?." https://bcghendersoninstitute.com/will-net-zero-get-us-to-net-zero-emissions-c9ae50a6e014. Accessed: 2022-03-22.

[3] E. R. Masanet, A. Shehabi, N. Lei, S. J. Smith, and J. G. Koomey, "Recalibrating global data center energy-use estimates," *Science*, vol. 367, pp. 984 – 986, 2020.

[4] "Sustainability in the cloud." https://sustainability.aboutamazon.com/environment/the-cloud?energyType=true. Accessed: 2022-03-23.

[5] I. Khan, "Temporal carbon intensity analysis: renewable versus fossil fuel dominated electricity systems," *Energy Sources, Part A: Recovery, Utilization, and Environmental Effects*, vol. 41, no. 3, pp. 309–323, 2019.

[6] D. S. Callaway, M. Fowlie, and G. McCormick, "Location, location, location: The variable value of renewable energy and demand-side efficiency resources," *Journal of the Association of Environmental and Resource Economists*, vol. 5, no. 1, pp. 39–75, 2018.

[7] M. J. Neely, "Stochastic network optimization with application to communication and queueing systems," *Synthesis Lectures on Communication Networks*, vol. 3, no. 1, pp. 1–211, 2010.

[8] "Regional carbon intensity forecast." https://data.nationalgrideso.com/carbon-intensity1/regional-carbon-intensity-forecast. Accessed: 2022-03-22.

[9] A. Eryilmaz, R. Srikant, and J. R. Perkins, "Stable scheduling policies for fading wireless channels," *IEEE/ACM Transactions on Networking*, vol. 13, no. 2, pp. 411–424, 2005.

[10] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," *IEEE transactions on automatic control*, vol. 37, no. 12, pp. 1936–1948, 1992.

[11] M. J. Neely, E. Modiano, and C. E. Rohrs, "Dynamic power allocation and routing for time-varying wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 1, pp. 89–103, 2005.

[12] C.-S. Yang, R. Pedarsani, and A. S. Avestimehr, "Communication-aware scheduling of serial tasks for dispersed computing," *IEEE/ACM Transactions on Networking (TON)*, vol. 27, no. 4, pp. 1330–1343, 2019.

[13] M. Aldossary and H. A. Alharbi, "Towards a green approach for minimizing carbon emissions in fog-cloud architecture," *IEEE Access*, vol. 9, pp. 131720–131732, 2021.

[14] A. Radovanovic, R. Koningstein, I. Schneider, B. Chen, A. Duarte, B. Roy, D. Xiao, M. Haridasan, P. Hung, N. Care, S. Talukdar, E. Mullen, K. Smith, M. Cottman, and W. Cirne, "Carbon-aware computing for datacenters," *IEEE Transactions on Power Systems*, pp. 1–1, 2022.

[15] Z. Abbasi, M. Pore, and S. K. Gupta, "Online server and workload management for joint optimization of electricity cost and carbon footprint across data centers," in *2014 IEEE 28th International Parallel and Distributed Processing Symposium*, pp. 317–326, 2014.

[16] M. R. Garey and D. S. Johnson, *Computers and intractability*, vol. 174. freeman San Francisco, 1979.

[17] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255, Ieee, 2009.

[18] "Google environmental report 2021." https://www.gstatic.com/gumdrop/sustainability/google-2021-environmental-report.pdf. Accessed: 2022-04-01.

[19] J. Malmodin and D. Lundén, "The energy and carbon footprint of the ict and e&m sector in sweden 1990-2015 and beyond," *ICT for Sustainability*, 2016.

[20] C. Luo, X. He, J. Zhan, L. Wang, W. Gao, and J. Dai, "Comparison and benchmarking of ai models and frameworks on mobile devices," *arXiv preprint arXiv:2005.05085*, 2020.

[21] M. A. B. Siddik, A. Shehabi, and L. Marston, "The environmental footprint of data centers in the united states," *Environmental Research Letters*, vol. 16, no. 6, p. 064017, 2021.

APPENDIX A
PROOF OF LEMMA 1

We first derive an upper bound on the sum of queue-length squares as follows:

$$\sum_{m=1}^{M} Q_m^{\mathrm{e}}(t+1)^2 + \sum_{m=1}^{M}\sum_{n=1}^{N} Q_{m,n}^{\mathrm{c}}(t+1)^2 \tag{27}$$

$$\leq \sum_{m=1}^{M} Q_m^{\mathrm{e}}(t)^2 + \sum_{m=1}^{M}\sum_{n=1}^{N} Q_{m,n}^{\mathrm{c}}(t)^2 + \sum_{m=1}^{M} a_m(t)^2 + \sum_{m=1}^{M}\left(\sum_{n=1}^{N} d_{m,n}(t)\right)^2 + \sum_{m=1}^{M}\sum_{n=1}^{N}\left(d_{m,n}(t)^2 + w_{m,n}(t)^2\right) + 2\sum_{m=1}^{M} Q_m^{\mathrm{e}}(t)\cdot\left(a_m(t) - \sum_{n=1}^{N} d_{m,n}(t)\right) + 2\sum_{m=1}^{M}\sum_{n=1}^{N} Q_{m,n}^{\mathrm{c}}(t)\cdot\left(d_{m,n}(t) - w_{m,n}(t)\right) \tag{28}$$

where (28) follows from (7), (8) and the inequality $(\max(a-b,0)+c)^2 \leq a^2+b^2+c^2+2a(c-b)$ for $a,b,c\geq 0$.

By rearranging equation (27) and (28), drift $\Delta(t)$ can be bounded as follows:

$$\Delta(t) \leq B + \sum_{m=1}^{M} Q_m^{\mathrm{e}}(t)\cdot\left(a_m(t) - \sum_{n=1}^{N} d_{m,n}(t)\right) + \sum_{m=1}^{M}\sum_{n=1}^{N} Q_{m,n}^{\mathrm{c}}(t)\cdot\left(d_{m,n}(t) - w_{m,n}(t)\right) \tag{29}$$

where B is a constant number defined in (18). By adding $VC(t)$ on both sides of (29) with some rearrangements, we finally conclude the proof.