DEMOTS: A Decentralized Task Scheduling Algorithm for Micro-Clouds with Dynamic Power-Budgets

Tharindu B. Hewage[1], Shashikant Ilager[2], Maria A. Rodriguez[1], Patricia Arroba[3], and Rajkumar Buyya[1]
[1]*The Cloud Computing and Distributed Systems (CLOUDS) Laboratory*,
School of Computing and Information Systems,
The University of Melbourne, Australia
[2]*Vienna University of Technology (TU Wien), Vienna, Austria*
[3]*Laboratorio de Sistemas Integrados (LSI), CCS–Center for Computational Simulation*
Universidad Politécnica de Madrid, Spain

2023 IEEE 16th International Conference on Cloud Computing (CLOUD) | 979-8-3503-0481-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/CLOUD60044.2023.00057

***Abstract*—The Internet of Things (IoT) driven latency-critical applications are deployed on lightweight Micro-Clouds at the network's edge. Renting physical space from geographically distributed colocation datacenters connected via a Wide Area Network (WAN) is a cost-effective way of deploying Micro-Clouds, despite WANs' dynamic communication latency from traffic congestion. However, this deployment approach can limit Micro-Clouds to operate within a soft power budget, as colocation datacenter providers utilize it to add more servers and lower capital costs through oversubscribing power infrastructure. As a result, Micro-Clouds use extreme energy reduction measures like power capping and task throttling to address power overdraw events, where power consumption exceeds soft power budget limits, which reduces the performance of latency-critical applications. We propose a solution where a dynamic power budget can be achieved by adding renewable energy sources to the existing soft power budget without upgrading power delivery systems. To take advantage of this, we propose a dynamic, decentralized task-scheduling algorithm called DEMOTS. DEMOTS effectively utilizes the available dynamic power budget in a WAN with varying degrees of network traffic congestion, thereby avoiding the need for extreme energy reduction measures. We implement DEMOTS on a simulation test-bed. Compared to state-of-the-art baseline using MCOP for decentralized task-scheduling in Micro-Clouds, DEMOTS reduces Power Overdraw Impact up to 19%, Task Latency Increase Impact up to 47%, and Task Schedule Time Impact up to 49%.**

***Index Terms*—Decentralized Resource Management, Micro-Clouds, Task Scheduling, Energy Efficient Computing**

I. Introduction

Motivation: The growing use of latency-critical applications in Micro-Clouds, driven by the Internet of Things (IoT), is expected to result in substantial energy demand. Micro-Clouds decentralize traditional hyper-scale clouds by dispersing computational capacity over a large number of lightweight data centers deployed at the network's edge [1], [2]. This approach exploits lower communication latency, making them ideal for IoT applications. Compared to traditional hyper-scale clouds, Micro-Clouds have a smaller energy footprint, resulting in smaller power ratings (less than three orders of magnitude than a traditional hyper-scale cloud [1]). However, due to the predicted fast growth rate, Micro-Clouds are expected to consume a similar amount of energy as traditional hyper-scale clouds by 2028 [3], [4].

Due to their lightweight nature, Micro-Clouds offer flexibility in cost-effective deployments. In this regard, deploying Micro-Clouds as tenants in multi-tenant colocation datacenters (hereby used as colocation datacenters) is widely adopted. For example, the datacenter operator Vapor IO plans to build thousands of edge colocation datacenters, which are a type of colocation datacenters hosting latency-critical IoT workloads [5]. In a colocation datacenter, the datacenter operator manages the datacenter facility and physical power/cooling infrastructures, and leases them to service providers (i.e. tenants) as a shared space. The service providers deploy Micro-Clouds on the leased resources. In this approach they only have to manage physical servers, thereby significantly reducing maintenance costs [5].

Keeping up with the rapid growth of Micro-Clouds is expensive for datacenter operators due to significant costs in building new datacenters [6]. Because of that, datacenter operators exploit utilizing existing resources. Typically, additional physical space is already available in datacenters. Therefore, the datacenter operators focus on employing server-level recovery mechanisms to oversubscribe its power infrastructure by provisioning additional servers. This technique is known as power oversubscription and it is a common approach used by data centers [5], [6].

In colocation datacenters, the datacenter operator cannot employ power oversubscription via server-level recovery mechanisms. This is because the colocation datacenter provider does not have control over the servers that are managed by tenants. To overcome this challenge, the datacenter operator implements a reduced *soft power budget* in the tenant's subscribed power. The difference between the *soft power budget* and the subscribed power capacity is then used to provision additional servers [5]. For example, in the case of a Micro-Cloud deployment as a tenant in a colocation datacenter, the Micro-Cloud operates under a reduced *soft*

2159-6190/23/$31.00 ©2023 IEEE
DOI 10.1109/CLOUD60044.2023.00057

power budget, such as 90% of the originally subscribed power. The remaining 10% of the power capacity contributes to the provision of additional servers.

However, operating under a *soft power budget* can result in Micro-Clouds having to resort to extreme power reduction measures such as power capping and workload throttling. Especially, when the rare power peaks exceed the available power budget (i.e. *power overdraw events*) [7]. This can have a detrimental impact on the performance of latency-critical workloads. Therefore, balancing between the power oversubscription techniques and the optimal workload performance is an ongoing challenge [6].

In this regard, our work aims to improve workload performance by minimizing *power overdraw events*. As a result, the need to employ extreme power recovery techniques is also minimized. One way to achieve this is increasing the *soft power budget* by combining additional energy. But drawing this additional energy from the power grid is not feasible because the power grids feeding energy to colocation datacenters are already stressed [8]. On the other hand, colocation datacenters already procure onsite renewable energy [9]. Therefore, drawing additional energy from on-site renewable sources is a viable option. The next challenge in this approach is combining additional energy with the *soft power budget* in a cost-effective manner. Such that expensive upgrades in the tenant power delivery system are minimized. When we consider an existing tenant in a colocation datacenter, this tenant subscribes to a certain amount of power. This subscribed power is delivered to the Micro-Cloud via a Power Delivery Unit (PDU) with a sufficient power capacity. When *soft power budget* is implemented, a portion of the PDU power capacity is left unused (e.g., for *soft power budget* equivalent to 90% of the subscribed power, a 10% in the PDU capacity is left unused). We identify that this underutilized PDU capacity can be used to combine additional energy with the *soft power budget*, thus avoiding expensive PDU upgrades. Therefore, we propose the addition of renewable power to the *soft power budget*, thereby creating a *dynamic power budget* for Micro-Clouds. The *dynamic power budget* increases the *soft power budget*, thereby minimising *power overdraw events* and the need for extreme power reduction measures. This improves workload performance under power oversubscription techniques being applied. An example of this approach is illustrated in Figure 1.

Furthermore, the *dynamic power budget* can be implemented across a geographically distributed network of Micro-Clouds deployed with colocation datacenters. In such a scenario, excess power in the *dynamic power budget* is highly likely to be available in one or more Micro-Clouds at a given time due to the intermittent nature of the renewable energy across different time zones. This motivates us to avoid employing extreme power reduction measures entirely, by offloading tasks to another Micro-Cloud with available power in its *dynamic power budget*. Concretely, if a Micro-Cloud is about to undergo a *power overdraw event*, its energy consumption can be reduced below its power budget by offloading a portion of its tasks to other Micro-Clouds that have available power in their *dynamic power budgets*. The feasibility of this approach is shown in Fig. 2 in which, the *dynamic power budget* is compared across three time zones. We observe that when a *power overdrawn event* occurs at one location, one or more of the other locations have available power in their *dynamic power budget*.

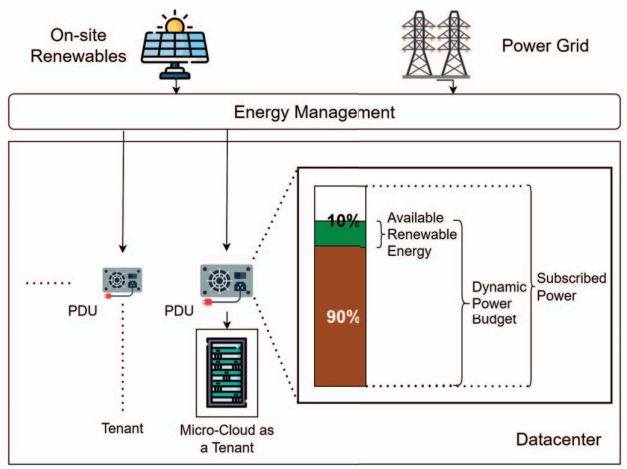

Fig. 1: Dynamic Power Budget

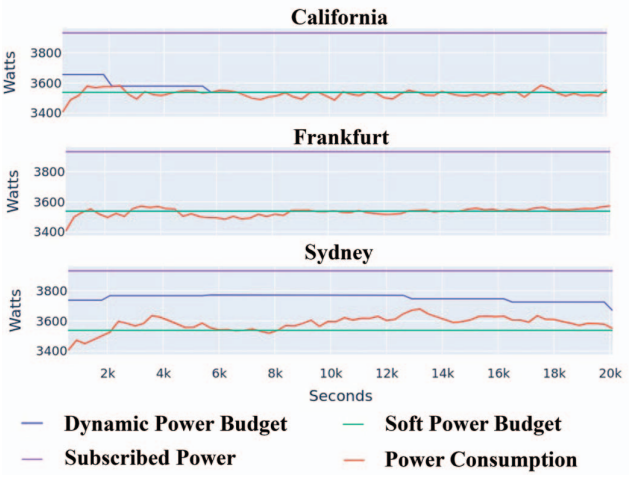

Fig. 2: Micro-Clouds with dynamic power budgets across different time zones (*Captured against Azure workload traces [15] and solar energy via the PVGIS tool [16]* over 5.5 hours of workload execution)

Challenges and Gaps: When tasks are offloaded across Micro-Clouds, it is carried out via dynamic task scheduling. Dynamic task scheduling allows for real-time adaptation to changes in the system, albeit at the cost of additional computational overhead, which can be minimized through lightweight computation approaches. Moreover, it is crucial to perform dynamic task scheduling in a decentralized manner. Otherwise, substantial round-trip communication times can lead to

TABLE I: A Comparison of Related Task Scheduling Algorithms

Work	Geo-Distributed	Dynamic	Decentralized	WAN traffic congestion-aware	Task latency-criticality-aware
Yuan et al. [10]	✓	✓	✗	✗	✓
Sharma and Rao [11]	✓	✓	✗	✗	✗
Zhao et al. [3]	✓	✓	✗	✗	✗
Sajid et al. [12]	✓	✓	✓	✗	✗
Tarneberg et al. [4]	✓	✗	✗	✗	✗
Selimi et al. [13]	✓	✓	✓	✗	✗
Panadero et al. [14]	✓	✓	✓	✗	✗
DEMOTS (This paper)	✓	✓	✓	✓	✓

delayed scheduling decisions. A few studies such as Multi-Criteria Optimal Placement (MCOP) [14] and lightweight service placement heuristic [13] have explored decentralized dynamic task scheduling in Micro-Cloud networks. They focus on multi-criteria optimization, such as node availability and the number of connections. But they do not consider the potential bottlenecks in inter-Micro-Cloud communication. Micro-Clouds are usually interconnected via Wide Area Networks (WAN). WAN can undergo severe traffic congestion, in which its tail latency can be worsened up to 2.5x [17]. Since Micro-Clouds frequently communicate with each other during decentralized task scheduling, dynamic WAN traffic congestion becomes a severe bottleneck to it. Additionally, these studies do not consider the energy optimization of Micro-Clouds. Consequently, we address these bottlenecks taken into account during scheduling decisions.

Our Work: We propose an approach to realize a *dynamic power budget* in a Micro-Cloud, and a novel task scheduling algorithm called DEMOTS: **De**centralized **M**ulti-criteria **O**ptimization **T**ask **S**cheduling to harness *dynamic power budget* across a network of Micro-Clouds. DEMOTS utilizes a decentralized approach to schedule tasks while tolerating dynamic WAN traffic congestion. DEMOTS outperforms state-of-the-art scheduling algorithms by up to 19% gain in reducing power overdraw impact, up to 47% gain in reducing task latency increase impact, and up to 49% gain in reducing task schedule time impact, across various tuning levels. In summary, the key contributions of our work are:

- An approach to realize a *dynamic power budget* for Micro-Clouds deployed in colocation datacenters
- A system model and performance metrics to measure the impact of *power overdraw events* on Micro-Clouds, and the impact of dynamic task scheduling on latency-critical IoT tasks.
- A formal definition of decentralized task scheduling with *dynamic power budgets*, and formulation of the multi-objective problem
- A novel dynamic decentralized task scheduling algorithm (DEMOTS) to solve the multi-objective problem by utilizing the *dynamic power budget* under realistic WAN traffic congestions.
- Extensive experiments and analysis of results comparing with the state-of-the-art algorithms demonstrating the superiority of DEMOTS.

The rest of the paper is organized as follows. In Section II we discuss the related literature. Section III provides the system model and its essential components. In Section IV we present our proposed DEMOTS algorithm. Section V describes the performance evaluation and experimental results. In Section VI we conclude and share future directions.

II. Related Work

Most existing approaches for dynamic task scheduling in multi-cloud networks, which are distributed geographically, focus on centralized renewable energy harnessing. Yuan et al. [10] focus on scheduling delay-tolerant applications in a centralized manner, while strictly meeting delay-bounded constraints, and taking into account the spatial and temporal variations of both grid and renewable energy. Sharma and Rao [11] propose a centralized scheduler that aims to optimize the percentage of renewable energy used. They identify a trade-off between average task waiting time and the percentage of renewable energy and suggest a method to improve the renewable energy percentage in geographically distributed multi-clouds. Zhao et al. [3] present a more recent approach that uses centralized deep reinforcement learning to utilize renewable energy in geographically distributed multi-clouds. Similarly, Tarneberg et al. [4] use a dynamic application placement technique to achieve the optimal placement of applications in mobile multi-cloud networks. However, this approach is not purely decentralized, and it is not capable of performing prioritized scheduling of specific criteria such as latency or power.

In contrast with centralized approaches, some recent works have explored decentralized approaches. Sajid et al. [12] use blockchain technology to implement a decentralized scheduling mechanism for energy management across geographically distributed multi-clouds. The use of renewable energy is managed through the blockchain network. Selimi et al. [13] employ a lightweight service placement heuristic that schedules tasks in community networks with dynamic network bandwidth and node availability. However, it requires the decision of a clustering parameter for optimum performance, which can be difficult in growing Micro-Clouds. To address this issue, Panadero et al. [14] propose the Multi-criteria Optimum Placement (MCOP) algorithm, which automatically handles the clustering issue. However, none of the approaches described above takes into account the impact on latency-critical tasks, as well as the impact of realistic WAN traffic congestion levels during their scheduling decisions.

Table I summarizes the related work compared to the proposed approach. The majority of the reviewed approaches use

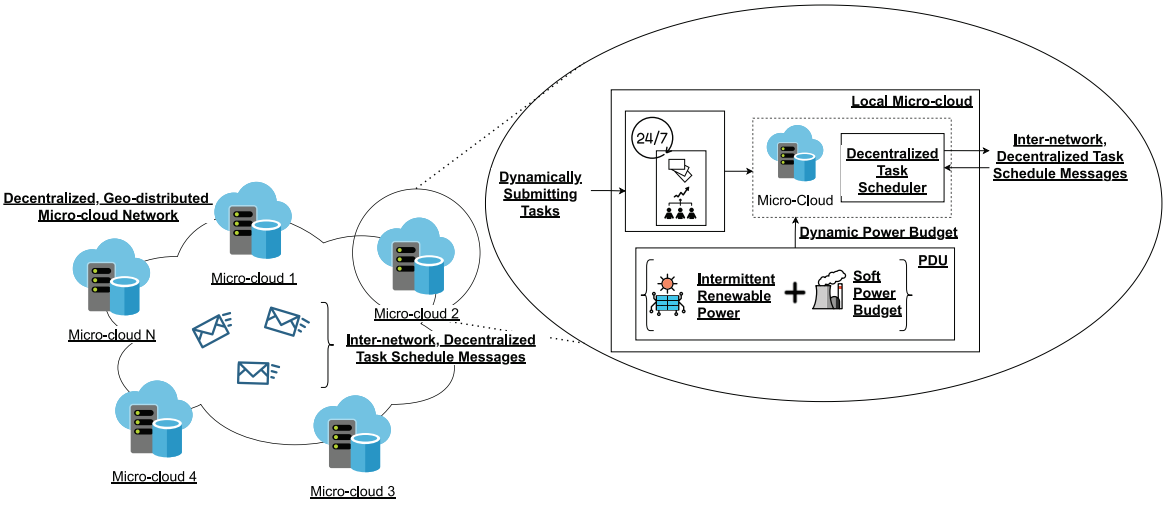

Fig. 3: System Architecture

centralized scheduling, which becomes a bottleneck for geographically distributed Micro-Cloud networks due to increased communication delays over WAN. Decentralized scheduling approaches provide an advantage, but none of them considers realistic WAN traffic congestion levels, energy optimizations, and the impact of scheduling decisions on the reliability of latency-critical IoT applications. The proposed approach aims to address these identified gaps.

III. System Model and Problem Formulation

A. System Model

The proposed system architecture model, shown in Figure 3, is a geographically distributed, decentralized, and homogeneous network of Micro-Clouds. The network latency between Micro-Clouds can change due to WAN traffic congestion. Each Micro-Cloud leases power infrastructure and physical space from a colocation data center provider. The provider employs the proposed approach to provide a *dynamic power budget* for Micro-Clouds. Users submit latency-critical IoT tasks dynamically to the closest Micro-Cloud. A decentralized task scheduler in each Micro-Cloud dynamically schedules tasks among the Micro-Clouds to avoid *power overdraw events*. The following subsections describe different models and performance metrics we use.

1) Workload Model: We use a location-aware, utilization-based workload model where the workloads are submitted dynamically to Micro-Clouds based on the local time and following a daily trend. The set of tasks that were submitted and executed by all the Micro-Clouds during the time period $t = 0$ to $t = T^{MAX}$ are denoted by Ω,

$$\Omega = \{t_1, ..., t_M\}$$

where M is the total number of tasks.

2) Power Consumption Model : Micro-Cloud power consumption is primarily determined by the power consumption of computing elements (i.e. power consumption of the Information Technology (IT) systems), and power overhead not used for computing (mostly used for cooling systems) [18]. Using the Power Usage Effectiveness (PUE) [19] metric, a linear relationship can be derived between the IT power and the overhead power [20].

$$P_{overhead} = (\text{PUE} - 1) \times P_{IT} \tag{1}$$

where P_{IT} is the IT power, $P_{overhead}$ is the power overhead, and PUE is a constant value. Due to this, we model power consumption just considering P_{IT}. For each Micro-Cloud P_{IT} is introduced as $Pc_i(t)$ (dynamic IT power consumption of i^{th} Micro-Cloud) using a host power consumption model. Our model is based on the CPU utilization level, as this resource represents the main contribution to the host power consumption [21].

$$Pc_i(t) = \sum_{j}^{hosts \in i^{th} \text{Micro-Cloud}} Pw_j(U_j(t)) \tag{2}$$

where $Pc_i(t)$ is the power consumption of the i^{th} Micro-Cloud and for the j^{th} host in that Micro-Cloud, the Pw_j is the power consumption model, and $U_j(t)$ is the CPU utilization at t^{th} time.

3) Dynamic Power Budget Model: To model the *dynamic power budget*, we combine the *soft power budget* (P_{spb_i}) and the intermittent renewable power provided by the colocation datacenter provider ($P_{rp_i}(t)$).

$$Pb_i(t) = P_{spb_i} + P_{rp_i}(t) \tag{3}$$

where $Pb_i(t)$ is the *dynamic power budget* of the i^{th} Micro-Cloud at the time t.

4) Power Overdraw Model: We model the power overdraw as the amount of the power consumption exceeding the $Pb_i(t)$ at t^{th} time.

$$P_{od_i}(t) = \begin{cases} Pc_i(t) - Pb_i(t) & \textit{if } Pc_i(t) > Pb_i(t) \\ 0 & \textit{otherwise} \end{cases} \quad (4)$$

where $P_{od_i}(t)$ is the power overdraw amount at the time t.

5) Performance Metrics: The following performance metrics are calculated for all Micro-Clouds in the network during the time period $t = 0$ to $t = T^{MAX}$.

- *Power Overdraw Impact (P_{OI}):* Measures magnitude and the time spent during *power overdraw events*.

$$P_{OI} = \sum_{i=1}^{N} \int_{t=0}^{t=T^{MAX}} P_{od_i}(t)\, dt \quad (5)$$

- *Task Schedule Time Impact (T_{SI}):* Measures the amount of time that tasks were blacked out during the dynamic scheduling.

$$T_{SI} = \sum_{t_k \in \Omega} T_{ST_{t_k}} \quad (6)$$

where $T_{ST_{t_k}}$ is the time spent by the task t_k during inter-Micro-Clouds scheduling.

- *Task Latency Increase Impact (L_I):* Measures the increased end-user communication latency and the amount of time spent with that.

$$L_I = \sum_{t_k \in \Omega} \sum_{i \in N} L_{t_{k_i}} \times T_{t_{k_i}} \quad (7)$$

where $L_{t_{k_i}}$ is the latency increase between the originating Micro-Cloud of the task t_k, and the i^{th} Micro-Cloud. $T_{t_{k_i}}$ is the time it spent in the i^{th} Micro-Cloud.

B. Problem formulation

The overall objective of our problem is to minimize Power Overdraw Impact (P_{OI}), but in doing so, also try to minimize both Task Schedule Time Impact (T_{SI}) and Task Latency Increase Impact (L_I).

IV. DEMOTS - DECENTRALIZED MULTI-CRITERIA OPTIMIZATION TASK SCHEDULING

We present our proposed DEMOTS approach, which is a decentralized algorithm that dynamically schedules tasks and adapts to traffic congestion levels in the inter-Micro-Cloud network. It runs in each Micro-Cloud.

DEMOTS continuously monitor for potential *power overdraw events* of the Micro-Cloud via the Power Overdraw Model ($P_{od_i}(t)$) defined in eq. 4. It tries to offload a batch of low-priority tasks to bring down the power consumption of the Micro-Cloud. For each task in the batch, it broadcasts a task offload request across the WAN and waits for responses within a fixed window of time using a time-out. Based on the responses received, the lexicographic method is used to filter and select a destination Micro-Cloud for each task. The selection is based on three optimization criteria designed to minimize Power Overdraw Impact (P_{OI}), Task Schedule Time Impact (T_{SI}), and Task Latency Increase Impact (L_I). Finally, the tasks are offloaded to the destination Micro-Clouds.

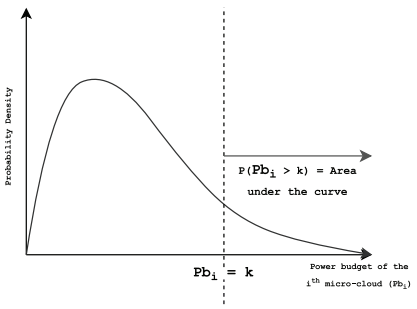

Fig. 4: Calculating Power Reliability (Pr_i) of the i^{th} candidate Micro-Cloud

Overall, DEMOTS propose three novel optimization criteria to minimize P_{OI}, T_{SI}, L_I, and a novel decentralized approach to offload tasks over WAN with dynamic traffic congestion levels.

A. Optimization Criteria

The following three optimization criteria are proposed for selecting a destination Micro-Cloud to offload the task t_k.

1) Minimize P_{OI}: Prioritizing Micro-Clouds with sufficiently available $Pb_i(t)$ would minimise the P_{OI}. We propose the Power Reliability (Pr_i) metric to measure the sufficiently available $Pb_i(t)$.

$$Pr_i(t) = P([Pb_i(t) - Pc_i(t)] > {Pc_i^{MAX}}_{t_k}) \quad (8)$$

Where ${Pc_i^{MAX}}_{t_k}$ is the maximum amount of power that the task t_k would consume at the i^{th} Micro-Cloud. We then estimate $Pc_i(t)$ with its rolling average value, denoted as Pc_{avg_i}.

$$Pr_i(t) = P([Pb_i(t) - Pc_{avg_i}] > {Pc_i^{MAX}}_{t_k})$$

$$Pr_i(t) = P(Pb_i(t) > ({Pc_i^{MAX}}_{t_k} + Pc_{avg_i}))$$

Define $k = {Pc_i^{MAX}}_{t_k} + Pc_{avg_i}$. Then,

$$Pr_i = P(Pb_i(t) > k)$$

where k is a positive constant. This simplifies Pr_i to the cumulative sum of the probability density function (PDF) of the Micro-Cloud's $Pb_i(t)$ as illustrated in Fig. 4. We estimate this PDF using historical data for each Micro-Cloud to calculate the Pr_i. Thus, the first optimization criterion is to select the candidate Micro-Cloud maximizing Pr_i.

$$\textbf{Criterion 1} : \arg\max_i(Pr_i) \quad (9)$$

TABLE II: Description of Symbols

Symbol	Description
Notation	*Description*
N	Number of Micro-Clouds in the inter Micro-Clouds network.
T^{MAX}	Time duration.
Ω	Set of tasks that were executed during T^{MAX}.
Hyper Parameter	*Description*
γ	Sensitivity of the dynamic task schedule initiation. A higher value leads to aggressive task offloading.
θ	Batch size of the low-priority tasks. A higher value initiates a larger number of task offload requests.

2) Minimize T_{SI}: Prioritizing Micro-Clouds with sufficient processing capability would ensure immediate task deployment, thus optimizing the T_{SI}. We propose the Redundant Processing Capacity metric of the i^{th} candidate Micro-Cloud (R_{pc_i}) to measure sufficient processing capability.

$$R_{pc_i} = \frac{C_{PE_i}(t) - T_{PE_{t_k}}}{T_{PE_{t_k}}} \tag{10}$$

In which,

- $C_{PE_i(t)=}$ *Available processing elements count in the* i^{th} *candidate Micro-Cloud at time* t
- $T_{PE_{t_k}}$ *= Minimum required processing elements for task* t_k

Thus, the second optimization criterion is to select the candidate Micro-Cloud maximizing R_{pc_i}.

$$\textbf{Criterion 2} : \arg\max_i(R_{pc_i}) \tag{11}$$

3) Minimize L_I: Prioritizing Micro-Clouds having the minimum increase in end-user communication latency would optimize the L_I. Thus, the third optimization criterion is to select the candidate Micro-Cloud minimizing the increase in end-user communication latency ($L_{increase_i}$).

$$\textbf{Criterion 3} : \arg\min_i(L_{increase_i}) \tag{12}$$

B. Decentralized Tasks Offloading over WAN

DEMOTS tasks offloading has three stages; `Initiate`: Broadcasts task offload requests of low-priority tasks, `Respond`: Responds to the task offload requests, and `Dispatch`: Selects the destination Micro-Cloud using three optimization criteria and offloads the task.

- `Initiate`: A Micro-Cloud periodically calculates its available power percentage relative to the Dynamic Power Budget ($Pb_i(t)$). If the available power percentage is below the threshold set by the tuning parameter $\gamma \in \{0, 1\}$, the Micro-Cloud selects a batch of low-priority tasks, where the batch size is set by the tuning parameter $\theta \in \{0, 1\}$. Since the scheduler does not have information about the priority level of the tasks, resource usage is used as an estimator of the priority, where high resource usage means high priority. For each task in the batch, the Micro-Cloud broadcasts a task offload request over the WAN, and a timeout is set to wait for responses. Due to WAN traffic congestion, obtaining responses from all available Micro-Clouds within a reasonable time cannot be guaranteed, thereby the timeout ensures a reasonable reaction time to *power overdraw events*. Upon meeting the time-out, a `Dispatch` stage is scheduled for each task offload request. This process is outlined in Algorithm 1.

Algorithm 1: Initiate

```
while each-refresh-interval do
    P_ratio(t) ← (Pb_i(t) − Pc_i(t)) / Pb_i(t);
    if P_ratio(t) ≤ γ then
        T_lp(t) ← getLowPriorityTasks(θ);
        while t_k ← T_lp(t) do
            broadcast(t_k);
            scheduleDispatch(t_k);
        end
    end
end
```

- `Respond`: Upon receiving a task offload request from another Micro-Cloud, a Micro-Cloud responds with the necessary information that is needed to compute three optimization criteria for the offloading task. This process is outlined in Algorithm 2, in which the processing capacity metric $C_{PE_i}(t)$ is calculated by the sub-routine $getPCM$, and Power Reliability (Pr_i) is calculated by the sub-routine $getPR$.

Algorithm 2: Respond

Data: Rq_{t_k}: *Offload request for* t_k
Result: Response
$L_{increase_i} \leftarrow getLatency(Rq_{t_k})$;
$C_{PE_i} \leftarrow getPCM(Rq_{t_k})$;
$P_{r_i} \leftarrow getPR(Rq_{t_k})$;
$replyBack(L_{increase_i}, C_{PE_i}, P_{r_i})$;

- `Dispatch`: When the time-out is reached for a task offload request, the destination Micro-Cloud is chosen by evaluating three criteria using the lexicographic method. The most significant criterion is Criterion 1, which ensures that Dynamic Power Budget ($Pb_i(t)$) is available. A subset of responses is filtered based on this criterion. Then, this subset is further filtered based on Criteria 2 and 3 to isolate a single response, which is selected as the destination Micro-Cloud. The task is then offloaded to this destination Micro-Cloud.

V. Performance Evaluation

In this section, we describe our experimental setup and demonstrate the effectiveness of DEMOTS algorithm by analyzing the results and comparing them with baselines.

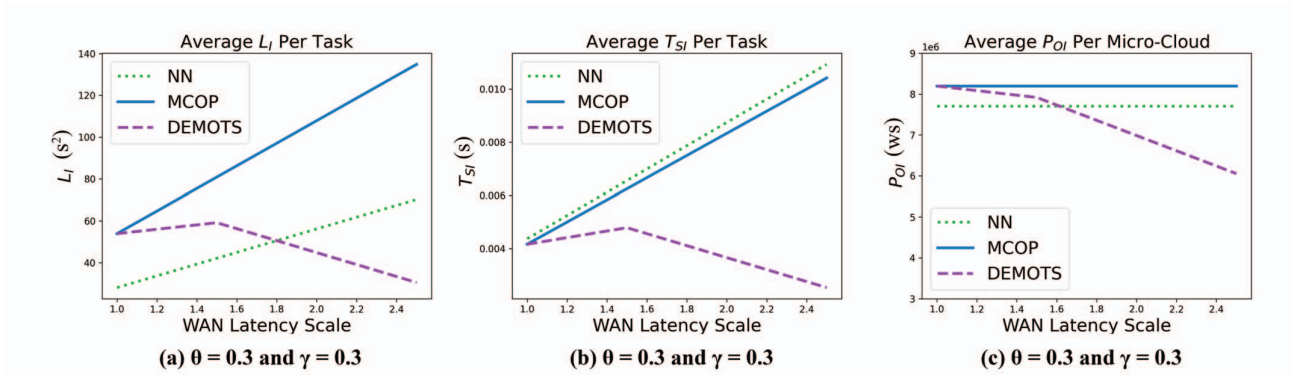

Fig. 5: Performance Metrics Comparison For θ = 0.3

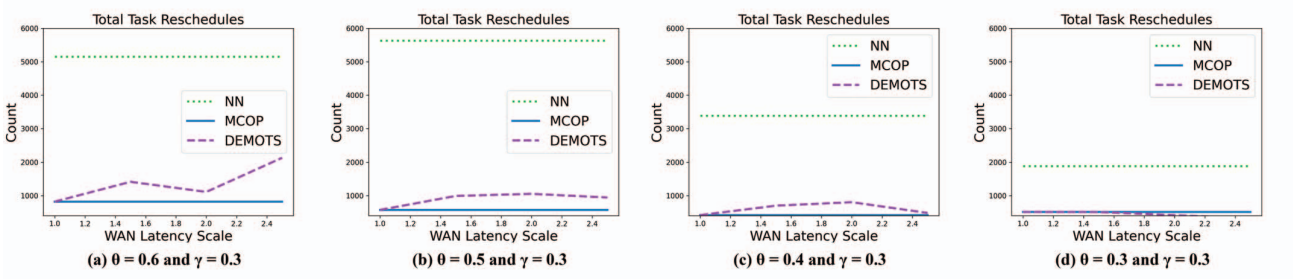

Fig. 6: Total Number Of Task Reschedules Against θ

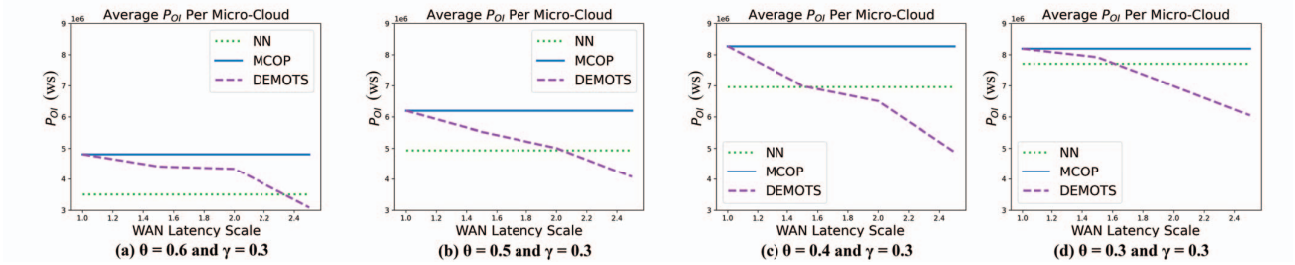

Fig. 7: Power Overdrawn Impact Performance Against θ

A. Experimental Setup

We evaluate our proposed solution in a simulated environment using the CloudSim toolkit [22]. We use real-world workload traces for generating application workload. We also configure other relevant parameters from real-world traces and models such as network latency data, power models, and solar energy traces.

We extend the Datacenter class of CloudSim toolkit to implement a power budget-aware Micro-Cloud component and further extend the Host class to implement power-aware hosts. Each host utilizes an extended PowerModel class to model the physical server that we use. We implement new Power Source classes and embed them into the extended Datacenter class to manage renewable energy through solar energy traces. The utilization of each VM and its associated workload is implemented using the extended Vm and Cloudlet classes of the CloudSim toolkit. To manage resource utilization data in the workload trace, we employ an extended Cloudlet Scheduler class. The inter-Micro-Cloud network is managed using the default implementation of the Network Topology class in the CloudSim toolkit, and network latency data is handled accordingly.

Micro-Cloud Design: We design a Micro-Cloud as a tenant residing in a colocation datacenter which occupies a full dedicated server rack [5]. We select the rack size as 42U, based on modern mixed-energy Micro-Clouds designs [23]. We consider a rack that has 21 Fujitsu RX300 S6 XeonE5620 servers, each occupying 2U. We use a power model developed specifically for the selected server type [24]. The peak power consumption of this server rack is 3.9 kW.

***Dynamic Power Budget* Implementation:** We implement our proposed *dynamic power budget* approach for the Micro-

Cloud. We set the *soft power budget* imposed by the datacenter operator at 90% of the Micro-Cloud's subscribed power (which is 3.9 kW, the peak power consumption in our Micro-Cloud design). Therefore, we provision a solar panel providing 265W peak power [23], to accommodate the remaining 10% of that. The solar energy is simulated using the real traces obtained from the European Commission's Photovoltaic Geographic Information System (PVGIS) [16], per each geographical location..

Dynamic Workload Submission: We utilize the Microsoft Azure public VM workload traces from the year 2019 [15]. It represents one of the most recent publicly available production VM workload traces [25], and exhibits a dynamic workload submission trend. We shift that trend based on the local time zone, such that the workload submission trend is location-aware.

Experiment Scenario: We employ a geographically distributed Micro-Clouds network. Each of the Micro-Cloud in the network is deployed in a colocation datacenter as a tenant. The colocation datacenters are located in dispersed geographical locations, such that they are in different time zones from each other. The geographical locations are configured based on AWS datacenter regions. While AWS regions are hyper-scale clouds, we assume our Micro-Clouds exist in similar locations inside colocated facilities to reflect the realistic scenarios, such as the presence of Micro-Clouds across time zones and geographically closer to application users. We use nine different AWS regions connected over a WAN, and the real average inter-region communication latency values [26]. In order to simulate the effects of WAN traffic congestion, we scale WAN communication latency from its average values, up to the worst-case upper bound, which is 2.5x [17].

B. Baseline Algorithms

We consider approaches suitable for Micro-Cloud networks and avoid comparisons with centralized scheduling approaches that can lead to significantly delayed scheduling decisions due to WAN traffic congestion. Therefore, we compare our DEMOTS approach with the following two decentralized scheduling methods.

- **Nearest Neighbour (NN)**: A heuristic that schedules tasks to the nearest available Micro-Cloud, in terms of the latency [27].
- **MCOP**: A dynamic decentralized task scheduling algorithm for Micro-Cloud networks [14]. To the best of our knowledge, MCOP is the state-of-the-art decentralized task scheduling algorithm for Micro-Clouds that provides faster execution based on its lightweight heuristic approach.

C. Results and Analysis

We carried out 24-hour-long experiments for different parameter configurations. The γ value determines between a reactive ($\gamma = 0$) and a proactive ($\gamma > 0$) approach towards managing *power overdraw events*. Our aim is to avoid these events entirely, thus we set γ at 0.3 (i.e., 30% of the available

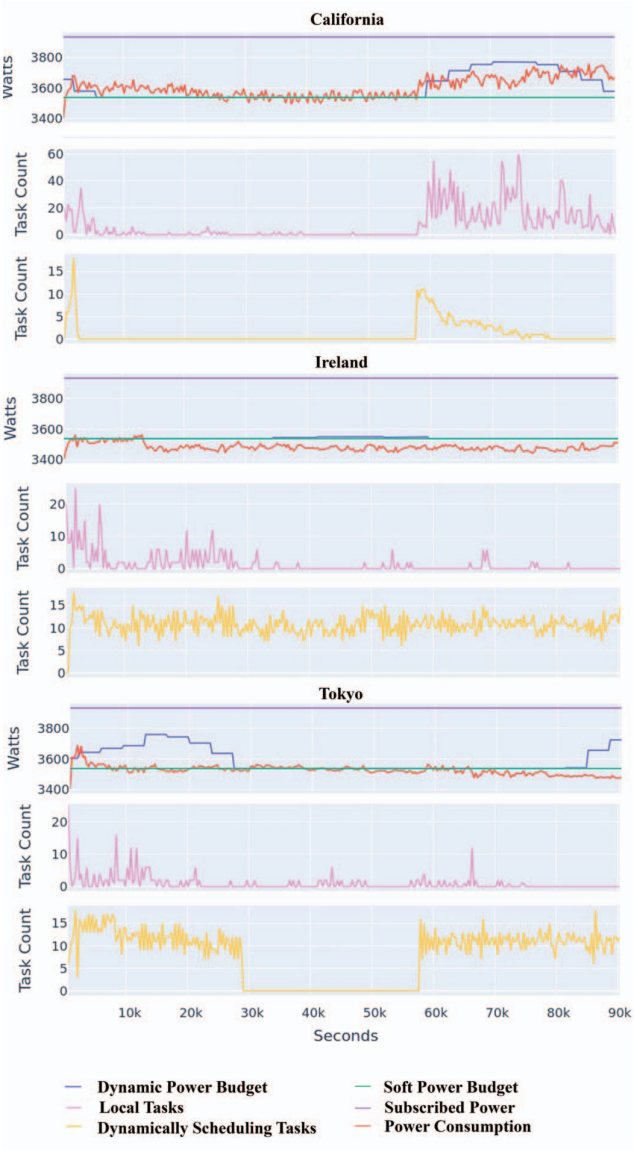

Fig. 8: DEMOTS: Harnessing Dynamic Power Budget (*Collected data from 24 Hours of workload execution*)

power triggers task offloading). The θ controls the batch size of tasks offloading. We observed scheduler sensitivity towards θ across a range of values ($\theta \in \{0.3, 0.4, 0.5, 0.6\}$). Each scenario was executed across worsening WAN traffic congestion by up-scaling communication latency. In all configurations, we set DEMOTS timeout, such that when the WAN latency values are at the average, DEMOTS is able to receive responses from all the Micro-Clouds in the network before offloading a task.

Fig. 5 shows scheduler performances for $\theta = 0.3$ in Task Latency Increase Impact (L_I), Task Schedule Time Impact (T_{SI}) and Power Overdraw Impact (P_{OI}). As WAN latency increases, both MCOP and NN show linear trends for L_I and T_{SI} (Fig. 5-a and Fig. 5-b), and a constant trend in

TABLE III: Performance Comparison of Schedulers

θ	Scheduling Algorithm	Mean Value across the WAN Latency Scale (1-2.5x)							
		Average P_{OI} per Micro-Cloud (ws)	Gain over MCOP (%)	Average L_I per task (s * s)	Gain over MCOP (%)	Average T_{SI} per task (s)	Gain over MCOP (%)	Total Number of Task Reschedules	Gain over MCOP (%)
0.3	NN	7704306.95	6.0%	51.52	47.91%	0.0080	-4.92%	1883	-267.77%
	MCOP	8195802.23	-	98.89	-	0.0076	-	512	-
	DEMOTS	7388672.81	9.85%	47.88	51.58%	0.0038	49.86%	448	12.5%
0.4	NN	6963946.19	15.85%	34.75	37.41%	0.0096	-123.35%	3384	-711.51%
	MCOP	8276334.03	-	55.52	-	0.0043	-	417	-
	DEMOTS	6665561.13	19.46%	37.50	32.45%	0.0040	5.41%	600	-44.06%
0.5	NN	4933240.99	20.46%	37.78	40.80%	0.0122	-101.75%	5629	-882.37%
	MCOP	6202747.05	-	63.83	-	0.00606	-	573	-
	DEMOTS	5202313.05	16.12%	38.53	39.63%	0.0054	9.70%	890	-55.41%
0.6	NN	3507866.21	27.06%	34.42	33.84%	0.010	-56.18%	5153	-528.41%
	MCOP	4809547.92	-	52.04	-	0.006	-	820	-
	DEMOTS	4162482.49	13.45%	36.86	29.15%	0.006	3.96%	1369	-67.0%

P_{OI} (Fig. 5-c). The reason behind both these trends is that MCOP and NN are not aware of the changes in WAN latency, thus performing the same scheduling decisions. Based on Equations 6 and 7, if the scheduling decisions remain the same, the L_I and T_{SI} change linearly with the increasing WAN latency, whereas based on Equation 5 the P_{OI} stays the same. In contrast, DEMOTS reacts to changing WAN latency and outperforms DEMOTS across all three metrics (Fig. 5-a, Fig. 5-b, and Fig. 5-c). This is justified as DEMOTS uses a timeout-based waiting approach, in which the worsening WAN latency forces it to change its scheduling decisions. DEMOTS achieve the same performance as MCOP when WAN latency is at its average. However, as WAN latency increases, DEMOTS take a different scheduling approach from MCOP, which converges in better performances.

Fig. 6 shows the comparison of the total number of task reschedules by schedulers, which is preferred to be minimized for latency-critical IoT tasks. Because for such tasks, reliability is critical, and an increased number of reschedules reduces task reliability. In that regard, NN shows significantly worst task reliability. NN's lowest number of task reschedules is achieved at $\theta = 0.3$, in which both MCOP and DEMOTS show significantly lower numbers of task rescheduled (Fig. 6-d). This trend continues for increasing θ (Fig. 6-c to Fig. 6-a). In comparison, DEMOTS and MCOP seem to be in a similar range, with DEMOTS having a slight increase over MCOP. The θ determines the task batch size, thus overall, decreasing θ results in a reduced number of task reschedules across all schedulers, as shown from Fig. 6-a to Fig. 6-d.

Fig. 7 shows scheduler sensitivity towards θ in optimizing Power Overdraw Impact (P_{OI}). In general, decreasing θ results in lowering P_{OI} performance (i.e., higher values for P_{OI}) for all schedulers as seen from Fig. 7-a to Fig. 7-d. NN has the worst sensitivity, with P_{OI} having a comparatively fast growth rate. In contrast, both MCOP and DEMOTS show better sensitivity. Moreover, DEMOTS outperform other schedulers by converging to better P_{OI} across the WAN latency scale, despite the decreasing θ.

To summarize, a comparison of average performance metrics over the WAN latency scale is depicted in Table III. While having better Power Overdraw Impact (P_{OI}) and Task Latency Increase Impact (L_I) gains over MCOP (6% to 27%, and 33% to 47% respectively), NN lags behind in Task Schedule Time Impact (T_{SI}) (-4% to -123%). Most importantly, NN needs to perform a higher number of task reschedules over MCOP (-267% to -882%), which significantly decreases task reliability. In contrast, DEMOTS performs much better in the number of tasks reschedules over MCOP (-67% to 12.5%). Therefore, both MCOP and DEMOTS surpass NN in scheduling latency-critical IoT tasks demanding the highest level of reliability. Moreover, DEMOTS outperforms MCOP by 36% to 47% in P_{OI}, 9% to 19% in L_I, and 3% to 49% in T_{SI}. Therefore, in the overall scheduling problem, DEMOTS is able to harness *dynamic power budget* to reduce *power overdraw events* by dynamically scheduling latency-critical IoT tasks over a WAN with dynamic traffic congestion. Fig. 8 showcase DEMOTS harnessing *dynamic power budget* across different time zones. In which, we observe the collective decentralized task offloading of DEMOTS successfully shares the workload based on available power. Concretely, the Micro-Cloud in California handles workloads near its power capacity (i.e., power consumption is around the *soft power budget*), until the observing time reaches 60k (seconds). Throughout this period of time, DEMOTS do not offload tasks to California. Afterwards, the zone receives excess *dynamic power budget*, in which California starts receiving tasks from other Micro-Clouds executing DEMOTS to utilize the excess power. The same behaviour can be seen in parallel for both Ireland and Tokyo.

VI. Conclusions and Future Work

Micro-Clouds leasing resources from power-oversubscribed colocation datacenters can operate under a *soft power budget*, in which rare *power overdraw events* are managed via extreme power overdraw recovery techniques affecting latency-critical IoT tasks. We proposed an approach to increase the *soft power budget* to a *dynamic power budget* with on-site renewable energy. To harness *dynamic power budget* for Micro-Clouds

deployed across Wide Area Network (WAN), we proposed DEMOTS: A dynamic decentralized task scheduling algorithm that tolerates dynamic WAN traffic congestion. Our extensive evaluation experiments show that the proposed DEMOTS outperforms the state-of-the-art decentralized scheduling algorithm by up to 19% reduction in Power Overdraw Impact, up to 47% reduction in Task Latency Increase Impact, and up to 49% reduction in Task Schedule Time Impact.

Future Work: The proposed approach utilizes a probability distribution model to predict solar energy based on historical solar energy data, while estimating the criticality of tasks based on resource usage. The accuracy of both techniques can be enhanced by incorporating deep learning-based prediction techniques. We plan to implement the proposed approach in a real colocated datacenter testbed to assess its effectiveness. In that, we plan to explore task re-scheduling with data, such as stateful tasks, in order to further evaluate DEMOTS' adaptability to scheduling under varying network performances.

ACKNOWLEDGMENTS

This work is partially supported by the University of Melbourne and the HiPEAC6 Network, financed by the European Union's Horizon2020 research and innovation program.

REFERENCES

[1] Q. Pei, S. Chen, Q. Zhang, X. Zhu, F. Liu, Z. Jia, Y. Wang, and Y. Yuan, "Cooledge: Hotspot-relievable warm water cooling for energy-efficient edge datacenters," in *Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, p. 814–829, 2022.

[2] M. Xu, Z. Fu, X. Ma, L. Zhang, Y. Li, F. Qian, S. Wang, K. Li, J. Yang, and X. Liu, "From cloud to edge: A first look at public edge platforms," in *Proceedings of the 21st ACM Internet Measurement Conference*, pp. 37–53, 2021.

[3] J. Zhao, M. A. Rodr̃iguez, and R. Buyya, "A deep reinforcement learning approach to resource management in hybrid clouds harnessing renewable energy and task scheduling," in *Proceedings of 2021 IEEE 14th International Conference on Cloud Computing (CLOUD)*, pp. 240–249, 2021.

[4] W. Tärneberg, A. Mehta, E. Wadbro, J. Tordsson, J. Eker, M. Kihl, and E. Elmroth, "Dynamic application placement in the mobile cloud network," *Future Generation Computer Systems*, vol. 70, pp. 163–177, 2017.

[5] M. A. Islam and S. Ren, "Ohm's law in data centers: A voltage side channel for timing power attacks," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, p. 146–162, 2018.

[6] A. G. Kumbhare, R. Azimi, I. Manousakis, A. Bonde, F. Frujeri, N. Mahalingam, P. A. Misra, S. A. Javadi, B. Schroeder, M. Fontoura, and R. Bianchini, "Prediction-Based power oversubscription in cloud platforms," in *Proceedings of the 2021 USENIX Annual Technical Conference (USENIX ATC 21)*, pp. 473–487, 2021.

[7] X. Fu, X. Wang, and C. Lefurgy, "How much power oversubscription is safe and allowed in data centers," in *Proceedings of the 8th ACM International Conference on Autonomic Computing*, p. 21–30, 2011.

[8] Y. Muhammad, R. Khan, M. A. Z. Raja, F. Ullah, N. I. Chaudhary, and Y. He, "Solution of optimal reactive power dispatch with facts devices: A survey," *Energy Reports*, vol. 6, pp. 2211–2229, 2020.

[9] P. Huang, B. Copertaro, X. Zhang, J. Shen, I. Löfgren, M. Rönnelid, J. Fahlen, D. Andersson, and M. Svanfeldt, "A review of data centers as prosumers in district energy systems: Renewable energy integration and waste heat reuse for district heating," *Applied Energy*, vol. 258, p. 114109, 2020.

[10] H. Yuan, J. Bi, and M. Zhou, "Spatiotemporal task scheduling for heterogeneous delay-tolerant applications in distributed green data centers," *IEEE Transactions on Automation Science and Engineering*, vol. 16, pp. 1686–1697, 2019.

[11] D. Sharma and S. Rao, "Scheduling computing loads for improved utilization of solar energy," *Sustainable Computing: Informatics and Systems*, vol. 32, p. 100592, 2021.

[12] S. Sajid, M. Jawad, K. Hamid, M. U. Khan, S. M. Ali, A. Abbas, and S. U. Khan, "Blockchain-based decentralized workload and energy management of geo-distributed data centers," *Sustainable Computing: Informatics and Systems*, vol. 29, p. 100461, 2021.

[13] M. Selimi, L. Cerdà-Alabern, F. Freitag, L. Veiga, A. Sathiaseelan, and J. Crowcroft, "A lightweight service placement approach for community network micro-clouds," *Journal of Grid Computing*, vol. 17, pp. 169–189, 2019.

[14] J. Panadero, M. Selimi, L. Calvet, J. M. Marquès, and F. Freitag, "A two-stage multi-criteria optimization method for service placement in decentralized edge micro-clouds," *Future Generation Computer Systems*, vol. 121, pp. 90–105, 2021.

[15] E. Cortez, A. Bonde, A. Muzio, M. Russinovich, M. Fontoura, and R. Bianchini, "Resource central: Understanding and predicting workloads for improved resource management in large cloud platforms," in *Proceedings of the 26th Symposium on Operating Systems Principles*, p. 153–167, 2017.

[16] J. R. C. European Commission, "Photovoltaic geographical information system." Available at https://re.jrc.ec.europa.eu/pvg_tools/en/DR. (2022).

[17] A. Saeed, V. Gupta, P. Goyal, M. Sharif, R. Pan, M. Ammar, E. Zegura, K. Jang, M. Alizadeh, A. Kabbani, and A. Vahdat, "Annulus: A dual congestion control loop for datacenter and wan traffic aggregates," in *Proceedings of the Annual Conference of the ACM Special Interest Group on Data Communication on the Applications, Technologies, Architectures, and Protocols for Computer Communication*, p. 735–749, 2020.

[18] M. Acton, P. Bertoldi, J. Booth, L. Newcombe, A. Rouyer, and R. Tozer, "2018 best practice guidelines for the eu code of conduct on data centre energy efficiency," *Publications Office of the European Union, Luxembourg, Tech. Report. EUR 29103 EN, 2018*, 2017.

[19] Z. Zhou, "Greenedge: Greening edge datacenters with energy-harvesting iot devices," in *Proceedings of the 27th IEEE International Conference on Network Protocols (ICNP)*, pp. 1–6, 2019.

[20] S. Pérez, P. Arroba, and J. M. Moya, "Energy-conscious optimization of edge computing through deep reinforcement learning and two-phase immersion cooling," *Future Generation Computer Systems*, vol. 125, pp. 891–907, 2021.

[21] S. Ilager, K. Ramamohanarao, and R. Buyya, "Etas: Energy and thermal-aware dynamic virtual machine consolidation in cloud data center with proactive hotspot mitigation," *Concurrency and Computation: Practice and Experience*, vol. 31, p. e5221, 2019.

[22] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, "Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Software: Practice and Experience*, vol. 41, pp. 23–50, 2011.

[23] R. Brännvall, M. Siltala, J. Gustafsson, J. Sarkinen, M. Vesterlund, and J. Summers, "Edge: Microgrid data center with mixed energy storage," in *Proceedings of the Eleventh ACM International Conference on Future Energy Systems*, p. 466–473, 2020.

[24] P. Arroba, J. M. Moya, J. L. Ayala, and R. Buyya, "Dynamic voltage and frequency scaling-aware dynamic consolidation of virtual machines for energy efficient cloud data centers," *Concurrency and Computation: Practice and Experience*, vol. 29, p. e4067, 2017.

[25] V. K. Jayakumar, S. Arbat, I. K. Kim, and W. Wang, "Cloudbruno: A low-overhead online workload prediction framework for cloud computing," in *Proceedings of the 2022 IEEE International Conference on Cloud Engineering (IC2E)*, pp. 188–198, 2022.

[26] A. Atrey, G. Van Seghbroeck, H. Mora, F. De Turck, and B. Volckaert, "Spech: A scalable framework for data placement of data-intensive services in geo-distributed clouds," *Journal of Network and Computer Applications*, vol. 142, pp. 1–14, 2019.

[27] H. Ma, Z. Zhou, and X. Chen, "Leveraging the power of prediction: Predictive service placement for latency-sensitive mobile edge computing," *IEEE Transactions on Wireless Communications*, vol. 19, pp. 6454–6468, 2020.