Measurement-Based Energy-Efficiency Optimization for On-demand Data Rate Provisioning in 5G RAN

Ramagiddaiah Eediga*, Preethi G V*, Anand SVR*, Chandra R. Murthy*, Ryan Husbands†
*Dept. of ECE, Indian Institute of Science, Bangalore, India. †British Telecom, London, UK.
{ramagiddaiah, preethigv, anandsvr, cmurthy}@iisc.ac.in, ryan.husbands@bt.com

***Abstract*—As network traffic grows, reducing energy costs and CO_2 emissions is essential for sustainable telecom operations. Energy efficiency (EE) is therefore a key design objective in 5G Radio Access Networks (RAN), where computing resources in the Central Unit (CU) and Distributed Unit (DU) must be managed while meeting Quality of Service (QoS) requirements. This work focuses on the CU/DU computing platform, a major contributor to RAN power consumption due to continuous baseband processing. We present a measurement-based framework that dynamically adjusts CPU core count and clock frequency to minimize the energy consumption while meeting the data rate requirements of User Equipment (UE). The proposed algorithm selects the most energy-efficient CPU configuration based on experimentally derived power–performance profiles from a practical 5G RAN testbed. Experimental evaluation on two RAN platforms shows power savings of up to 16% and 35%, respectively, compared to a baseline configuration with all CPU cores active at the maximum base clock frequency. The results demonstrate the effectiveness of the proposed framework for energy-efficient operation in practical 5G RAN deployments.**

***Index Terms*—5G RAN, Energy Efficiency, Open RAN, vRAN, CPU Frequency Scaling**

I. Introduction

Software/computational energy efficiency (EE) plays a foundational role in the transition to green networks, offering a cost-effective way to reduce the carbon footprint of 5G networks [1]. Globally, key industry alliances and standardization bodies (e.g., NGMN [2], NextG, ETSI, ITU [3], IEEE [4], 3GPP [5], O-RAN [6]) collaborate to evolve means to reduce energy consumption in Information and Communication Technologies (ICT). This paper describes a structured, measurement-based approach for adaptively configuring computational resources that minimize the power consumption of the base station (gNB), with emphasis on meeting the required data rates for end-user 5G applications.

A. Key Contributions

Our key contributions are as follows:

1) Instrumentation for energy measurements in a physical over-the-air (OTA) 5G Radio Access Network (RAN).
2) Characterizing the energy consumption behavior based on the measurements. Identification of the key system configuration knobs affecting the energy consumption.
3) Developing a lookup table based procedure for optimizing the energy consumption using the measured data.
4) Quantifying the energy savings obtainable by optimizing key RAN parameters.
5) Developing an O-RAN compliant closed-loop software automation for energy monitoring and control using open source Linux tools.

The takeaway from our study is that the CPU clock frequency and the number of active CPUs are the key parameters that determine a gNB's energy consumption. Choosing them optimally based on the network traffic demand results in significant energy savings without compromising on the performance.

B. Related work

Reducing the power consumption of 5G networks is a key focus in the recent literature. Several strategies have been proposed, including sleep mode management [7], computational resource optimization [8], and machine learning-based methods [9]. In line with 3GPP and O-RAN standards [10], these approaches employ tactics such as carrier shutdown, deep sleep modes, and AI-driven solutions [11].

In [12], the authors proposed dynamic CPU sleeping, a scheduling algorithm to improve EE in the Platform for Open Wireless Data-driven Experimental Research (POWDER) testbed, reporting energy savings up to 22%. The work in [13] investigates energy efficiency improvement in virtualized RAN (vRAN) servers by dynamically adjusting CPU frequency based on sub-second workload variations. It uses minimal interface information and low resource overhead, leveraging CPU frequency scaling and Linux eBPF for deadline monitoring, achieving up to 45% CPU power savings. In contrast, this work experimentally characterizes the power–performance behavior of CPU core and frequency settings in a practical 5G RAN deployment.

In [14], we reported preliminary observations on gNB power consumption. This work extends those results through systematic power–performance characterization and lookup-table-based optimization. Further, we adopt a measurement-driven approach to identify energy-efficient operating configurations in a practical 5G RAN testbed, unlike existing approaches that often rely on analytical models or simulations.

C. Functional Overview

The end-to-end 5G testbed deployed in our lab, shown in Fig. 1, offers a fully functional 5G gNB connected to commercial 5G phones serving as user equipment (UE). The network is implemented using the OpenAirInterface (OAI) 5G RAN and core platform [15]. The core handles UE authentication, session management, and data forwarding to

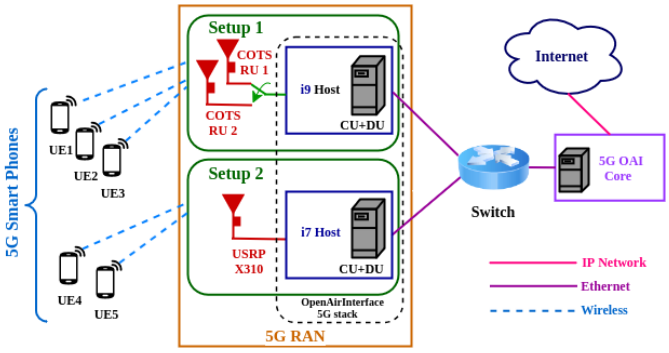


Fig. 1: The 5G testbed architecture used in this work.

external networks, while the gNB manages over-the-air communications, radio resource allocation, and UE connectivity to the core network. The testbed runs on an Intel-based system with Ubuntu 22.04 OS. Power consumption across different configurations was systematically evaluated using a smart energy meter for real-time monitoring.

The power consumption experiments were conducted using two different configurations. **Setup 1** features two Commercial-off-the-shelf (COTS) RUs labeled as COTS RU1 and COTS RU2. In Setup 1, the 5G RAN is configured with 100 MHz channel bandwidth, 30 kHz subcarrier spacing (SCS), 273 Physical Resource Blocks (PRB), and operates on band n78 using an Intel Core i9 host platform. In **Setup 2**, the CU/DU runs on an Intel platform connected to a USRP X310 software-defined radio (SDR) unit. The 5G RAN is configured with 40 MHz bandwidth, 30 kHz SCS, band n78, and employs 106 PRBs. The RUs support power consumption monitoring and control of its modes of operation.

D. Background work

Baseline measurements show that the RU consumes about 37 W in idle mode, increasing by 4–5 W when transmit power rises from -10 dBm to 24 dBm. Switching from SISO to 2×2 MIMO increases power by approximately 1–2 W with limited throughput gain in our setup.

Measurements at UE distances of 1–10 m showed RSRP decreasing from -78 dBm to -92 dBm, while the CU/DU power consumption and throughput remained largely unchanged. The UEs operated at the highest MCS throughout this range. This indicates that CU/DU power consumption is insensitive to the UE distance in the tested indoor deployment.

The RU consumed $\sim$37 W during uplink (10–50 Mbps) and $\sim$38 W during downlink (up to 300 Mbps), regardless of distance. The CU/DU (Setup 1, Fig. 1) increased power from 115 W to 125 W as the MCS index increased from 9 to 27, reflecting UE states from idle to peak downlink throughput. These observations indicate stable performance within 10 m, with throughput degradation beyond that range [16].

The CU/DU power consumption is significantly higher than that of the RU. For example, the CU/DU consumes about 125 W (100 MHz, i9) and 61 W (40 MHz, i7), while the RU consumes 44–51 W. This motivates optimizing CU/DU power consumption by selecting appropriate CPU core count and clock frequency, which have the highest impact on power consumption among the parameters considered.

We next present our workflow for selecting energy-efficient CPU configurations while meeting the data rate requirements.

II. Workflow and Methodology

Our goal is to determine the optimal CPU core count and clock frequency for maximizing energy savings in the RAN without compromising on the performance.

We consider a *baseline configuration* in which the CU/DU operates with all CPU cores active at the maximum clock frequency, representing a conservative provisioning strategy for real-time baseband processing. All energy savings are reported relative to this baseline. This is also the default mode of operation of OAI.

Disabling 4 of the 8 performance cores (P-cores) in the BIOS reduced power consumption by 16 W (21 W) when the gNB was idle (during DL activity), indicating that fewer active P-cores can significantly reduce power usage. Thus, we need to know what data rates can be supported and the corresponding power consumption across the system parameter settings. We characterize this by sweeping across all available CPU cores and supported clock frequencies at the gNB, evaluating the impact on power consumption and data rate.

Algorithm 1 Data collection and lookup table creation

Data collection:
$CPU_{\text{array}} \leftarrow$ All available CPU configurations
$Freq_{\text{array}} \leftarrow$ All available clock frequencies
for each $C_j \in CPU_{\text{array}}$ **do**
  for each $f_k \in Freq_{\text{array}}$ **do**
    Run the base station
    Record power consumption $P(C_j, f_k)$ and data rate $Rate(C_j, f_k)$
  end for
end for

Lookup table creation:
Initialize $Rate_{\text{dict}} \leftarrow$ empty dictionary
for each rate R_i in steps of 5 Mbps **do**
  $P_{i,min} \leftarrow Power(C_{max}, f_{max})$
  for each $C_j \in CPU_{\text{array}}$ **do**
    for each $f_k \in Freq_{\text{array}}$ **do**
      if $Rate(C_j, f_k) \geq R_i$ **and** $P(C_j, f_k) \leq P_{i,\min}$ **then**
        $Rate_{\text{dict}}[R_i] \leftarrow (C_j, f_k, P(C_j, f_k))$
        $P_{i,\min} \leftarrow P(C_j, f_k)$
        break
      end if
    end for
  end for
end for

We present two algorithms. **Algorithm 1**, *Data collection and lookup table creation:* We characterize the system by measuring the power consumption using a smart energy meter and the data rate using the `iperf3` test for each value

of the number of CPU cores in CPU_{array} and each CPU clock frequency in $Freq_{array}$. UDP traffic was generated using `iperf3` between the UE and an `iperf3` server. For each configuration, the maximum stable throughput (i.e., the measured throughput matched the `iperf3` traffic generation rate for at least 5 minutes) and the corresponding average power consumption were recorded and used to construct the lookup table. Then, we create a lookup table for the optimal configuration for each data rate, starting from 5 Mbps up to the maximum throughput the gNB can support, in steps of 5 Mbps. **Algorithm 2**, *Finding optimal configuration*: For a given required data rate R_{req}, we find the optimal number of CPU cores CPU_{opt} and the corresponding frequency $Freq_{opt}$.

Algorithm 2 Optimal configuration for a required rate R_{req}

1: $i \leftarrow \{\, m \mid Rate_{\text{dict}}[R_{m-1}] < R_{\text{req}} \leq Rate_{\text{dict}}[R_m] \,\}$
2: $CPU_{\text{opt}} \leftarrow Rate_{\text{dict}}[R_i].C$
3: $Freq_{\text{opt}} \leftarrow Rate_{\text{dict}}[R_i].f$
4: Operate the base station CPU using CPU_{opt} and $Freq_{\text{opt}}$

Our objective is to minimize power consumption while ensuring that the required throughput is achieved. Since CPU power consumption increases monotonically with both the number of active cores and operating frequency, selecting the lowest-energy CPU configuration that satisfies the data rate requirement leads to an energy-efficient operating point. By empirically characterizing the relationship between achievable data rate and CPU configurations, the proposed approach identifies configurations that approximate the minimum-power solution under the throughput constraint. Finally, we note that while the exact savings are platform-dependent, the methodology can be used to identify energy-efficient operating points in any setup.

Computational Complexity: The proposed algorithm uses a precomputed lookup table to map data rates to CPU configurations, enabling constant-time ($O(1)$) runtime selection of CPU core count and clock frequency.

In practice, the gNB knows the data rate requirement of each flow during data radio bearer (DRB) setup and adapts to traffic changes by modifying, terminating, or creating DRBs. Since data rate variations occur over a timescale of hundreds of milliseconds while CPU frequency updates take only a few milliseconds, the proposed lookup table-based approach enables rapid adaptation with minimal overhead.

A. Technical Challenges

In this section, we discuss some of the practical challenges that impact the data collection, measurement accuracy, and repeatability of the results, and how we addressed them.

- Real-time RAN implementations prioritize deterministic processing latency over aggressive CPU power management. For example, OAI recommends configuring the CPU in *performance* mode and disabling CPU deep sleep states to avoid PHY processing jitter. Our baseline configuration follows this practice, with all CPU cores active and operating at the maximum CPU clock frequency.

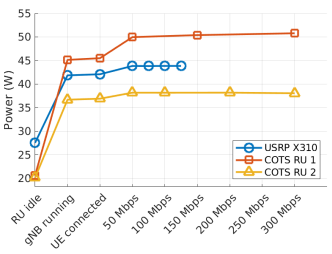


Fig. 2: Power consumption of USRP X310 and COTS RUs.

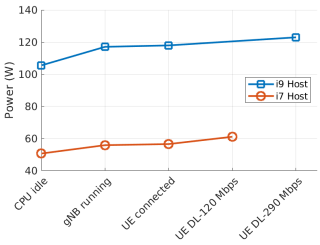


Fig. 3: CU/DU power consumption for Intel i9 & i7 setups 1 & 2.

- Due to the absence of built-in firmware support in the RU to monitor its real-time power usage, we had to use external hardware tools, such as a smart energy meter, to accurately measure power consumption.
- CU/DU power measurements showed fluctuations caused by Turbo Boost's dynamic frequency scaling. To ensure stable/repeatable measurements, this feature was disabled during the experiments. Thus, built-in power management tools in Linux are unsuitable for RAN deployments.
- The study of the effect of power consumption on the number of UEs called for capturing small power variations in the overall system. On the RU side, the fidelity of the smart energy meter did not allow us to record small fluctuations in the power consumption. On the CU/DU side, we were able to capture even small fluctuations in power usage using the *Scaphandre* tool.
- BIOS-based CPU core control proved tedious for repeated experiments. We therefore used Linux runtime core control to dynamically enable or disable CPU cores based on network load (e.g., `echo 0 | sudo tee /sys/devices/system/cpu/cpu1/online`).
- With recent CPU drivers like *intel_pstate* supporting only performance and powersave governors, effective system power control was limited. We resolved this by switching to *acpi-cpufreq* and using the *userspace* governor for explicit CPU frequency control.

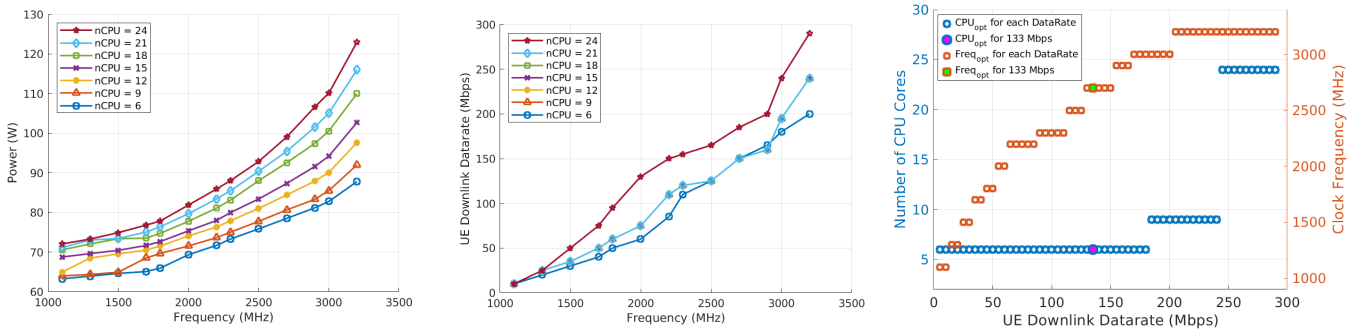
III. Numerical Results

We present power consumption measurements of the RAN, including the RU and CU/DU software protocol stack, and the resulting energy savings relative to the baseline configuration.

A. RU Power Consumption

Our RUs exhibited results as displayed in Fig. 2, with half of the power being consumed when the RU is idle, a significant increase when the gNB starts running, and a slight further increase when the UE is connected but idle. Downlink data transmission to the UE has minimal impact on overall power consumption with variations in UE throughput. Active user data transfer therefore contributes only marginally to the overall power usage of the RU.

Each RU supports a different maximum throughput based on its supported bandwidth. The USRP X310 showed a maximum power consumption of ∼45 W while supporting a downlink



(a) Power vs. CPU frequency for different core counts.

(b) UE downlink rate vs. CPU frequency for different core counts.

(c) Optimal CPU configuration across data rates (highlighted for $R_{req} = 133$ Mbps).

Fig. 4: Power and downlink rate performance versus CPU frequency, and optimal CPU configuration for Setup 1.

throughput of 115 Mbps using 40 MHz bandwidth. COTS RU1 and RU2 consumed around 51 W and 38 W, respectively, and supported a throughput of up to 300 Mbps using a 100 MHz SISO configuration.

B. CU/DU Power Consumption

Figure 3 shows the power usage across gNB states. The i9 host's power consumption varies from 105–123 W, while the i7 host ranges between 50–61 W. Both hosts display similar power consumption behavior, increasing during active transmission and decreasing during idle or low-traffic conditions.

Our experimental evaluation on the i9 host (see Fig. 4), measured the average power consumption and DL throughput across CPU frequencies ranging from 1100 to 3200 MHz, with CPU core counts varying from 6 to 24 in increments of 3. The data, collected as described in **Algorithm 1**, is plotted in Fig. 4a and Fig. 4b. For the i7 host, CPU frequencies ranged from 1300 to 3000 MHz, with core counts of 1, 2, 3, 5, and 8. The corresponding results are shown in Fig. 5a and Fig. 5b.

Fig. 4a shows that the power consumption increases roughly quadratically with the clock frequency. Increasing the number of CPU cores leads to higher power consumption. However, this dependency is not amenable to a simple polynomial fit.

Fig. 4b shows that the data rate increases roughly linearly with the clock frequency. However, the data rate remains nearly constant as the number of CPU cores is increased, except for the cases of 6 and 24 CPUs. This is because the baseband processing pipeline is limited by early stages of the processing chain, preventing additional CPU cores from improving throughput beyond a certain point. Since using 9 through 21 cores achieves the same data rate at each CPU frequency while the 9-core configuration consumes the least power, CPU configurations with 12, 15, 18, and 21 cores can be excluded. The cases with 6 and 9 CPU cores can support a maximum throughput of 200 and 240 Mbps, respectively, when operating at 3200 MHz. Throughput requirements above 240 Mbps can only be met by using all 24 CPU cores. Overall, the dependence of the data rate on the number of CPUs and clock frequency did not admit a simple formula, and led to our solution of constructing a lookup table to map the required data rate to the number of CPU cores and clock frequency.

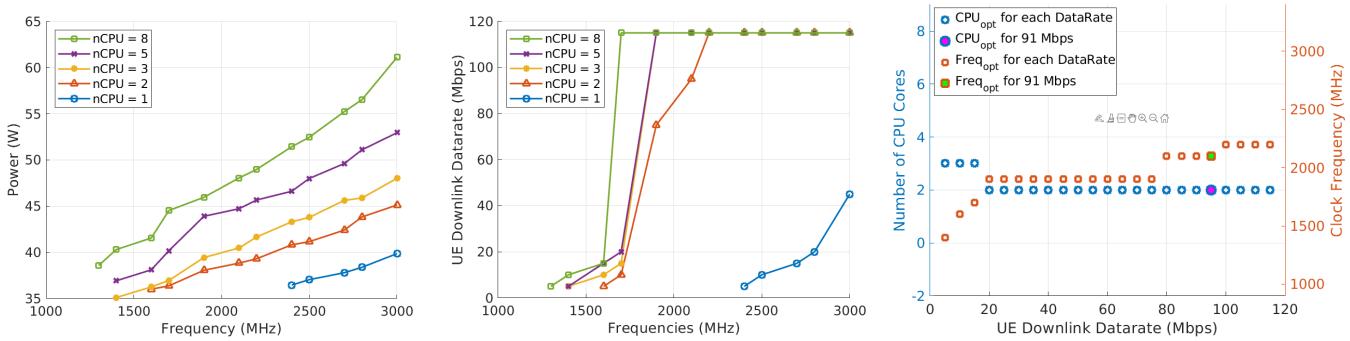
Fig. 4c plots the lookup table determined by **Algorithm 1**. For data rates up to 180 Mbps, 6 CPU cores with an appropriate clock frequency is optimal, while 9 CPU cores is optimal between 185 Mbps and 240 Mbps. For data rates from 245 to 290 Mbps, 24 CPU cores and a clock frequency of 3200 MHz is required, i.e., the system has to be operated at its full capacity. The solid markers in the figure are an example of the optimal configuration when $R_{\text{req}} = 133$ Mbps. Overall, across different R_{req} up to 240 Mbps, the optimal configuration results in power savings of at least $\sim$16%.

Fig. 5 presents the experimental results for the i7 host. Power consumption increases approximately linearly with clock frequency for different active core counts (Fig. 5a). Although the curves differ slightly from the i9 host, the overall trend of higher power usage with increased frequency and active cores holds. From Fig. 5b, the maximum throughput is 115 Mbps. The optimal configurations obtained using **Algorithm 1** are shown in Fig. 5c. A difference compared to Setup 1 is that for data rates up to 15 Mbps, the 2-core configuration operating at 1900 MHz or higher is not energy-optimal, as it consumes more power than the 3-core configuration with lower clock frequencies. Beyond 15 Mbps, the optimal configuration uses 2 active cores and higher clock frequencies with increasing data rates, since increasing frequency incurs a smaller power increase than activating an additional core.

C. Quantifying Energy Savings

We consider Setup 1 of Fig. 1 to quantify the percentage energy savings by using **Algorithm 2** relative to the baseline configuration. We consider average $R_{\text{req}} = \{40, 80, 120, 160, 200, 240\}$ Mbps. To model bursty traffic patterns, we use an ON–OFF Pareto traffic model, which effectively captures large bursts and the heavy-tailed behavior observed in real-world network traffic. Table I presents the percentage of energy savings for the above-mentioned average data rates under two different load scenarios:

- Fixed Load – Stable traffic with the gNB operating continuously at CPU_{opt} and $Freq_{\text{opt}}$.



(a) Power vs. CPU frequency for different core counts.

(b) UE downlink rate vs. CPU frequency for different core counts.

(c) Optimal CPU configuration across data rates (highlighted for $R_{req} = 91$ Mbps).

Fig. 5: Power and downlink rate performance versus CPU frequency, and optimal CPU configuration for Setup 2.

- Variable Load – Bursty traffic with the same average data rate, requiring dynamic adaptation.

TABLE I: Energy savings (%) for different load scenarios.

Average data rate	**Energy savings (%)**	
(Mbps)	**Fixed Load**	**Variable Load**
40	47.07	45.45
80	41.69	40.47
120	38.98	33.65
160	33.98	28.57
200	28.61	25.32
240	25.18	24.38
290	0	21.12

As the traffic load approaches system capacity, the gNB must operate close to the highest CPU frequency and core configuration, thereby reducing the potential for energy savings. For average loads of 40 and 80 Mbps, energy savings are nearly the same in both scenarios, as energy consumption varies linearly with data rate in this range, making it insensitive to small variations when system configuration is adjusted dynamically. However, at 120 and 160 Mbps, fixed load achieves ∼5% higher savings than variable load, due to the more frequent use of higher configurations under variable load. At 200 and 240 Mbps, this gap narrows as power consumption in the fixed load case also increases and approaches that of the highest configuration. At 290 Mbps, the fixed-load scenario achieves 0% energy savings since the gNB operates continuously at its maximum configuration. In contrast, the variable-load scenario sill achieves ∼21% energy savings because the gNB runs at peak configuration only when demand reaches or exceeds 290 Mbps, and scales down during lower-demand periods to reduce energy consumption.

The experiments were conducted with a limited number of UEs; larger-scale multi-UE studies are left for future work.

IV. Conclusions

In this work, we developed strategies to minimize gNB power consumption by dynamically adjusting the number of active CPU cores and system clock frequencies, while meeting user data requirements. In particular, using minimal cores and lower clock rates yielded measured savings of ∼16% in Setup 1 while supporting data rates up to 240 Mbps. Applying the same approach in Setup 2 resulted in even greater savings of about 35% for data rates up to 115 Mbps.

These gains were achieved while satisfying the required throughput targets, as the gNB anticipates data demand and adjusts parameters tens of milliseconds in advance. Future work will investigate latency-related trade-offs under rapidly varying traffic conditions and additional RAN energy-saving mechanisms such as capacity cell switch-off.

Acknowledgment

Financial assistance for this work from British Telecom, UK and MeitY, Govt. of India, is gratefully acknowledged.

References

[1] International Energy Agency, “Energy Efficiency,” 2023.
[2] NGMN Alliance, “Green Future Networks: A Roadmap to Energy Efficient Mobile Networks,” NGMN Publication, Jul. 2024.
[3] ITU-T, “Recommendation L.1331: Methodologies for Evaluating Energy Efficiency of Mobile Networks,” ITU, 2020.
[4] IEEE, “IEEE Sustainable ICT Initiative,” 2023.
[5] 3rd Generation Partnership Project (3GPP), “Study on network energy savings for NR,” Technical Report TR 38.864 V18.1.0, Mar. 2023.
[6] O-RAN Alliance, Sustainability Focus Group, “Potential Energy Savings Features in O-RAN,” White Paper, Jan. 2025.
[7] V. Kapustynska, “Energy-Saving Techniques in the Next Generation of Mobile Communication Networks,” MDPI ECP, May 2023.
[8] V. S. Rao and G. R. Babu, “Improved Resource Allocation in 5G C-RAN using Hybrid Swarm Intelligence,” in *IEEE IATMSI*, Mar. 2025.
[9] J. K. F. Okyere, “An Overview of Techniques for Enhancing Energy Efficiency in 5G Networks,” in *IEEE ICCSC*, 2024.
[10] O-RAN Alliance, “O-RAN Architecture Description v13.0,” Feb. 2025.
[11] R. Tan, Y. Shi, Y. Fan *et al.*, “Energy Saving Technologies and Best Practices for 5G Radio Access Network,” *IEEE Access*, 2022.
[12] S. Urumkar, B. Ramamurthy, and S. Sharma, “Improving Energy Efficiency In Open RAN Through Dynamic CPU Scheduling,” in *2023 IEEE ANTS*, 2023.
[13] A. Kalia, N. Lazarev, L. Xue *et al.*, “Towards Energy Efficient 5G vRAN Servers,” USENIX, Apr. 2025.
[14] E. Ramagiddaiah, G. V. Preethi, S. V. R. Anand *et al.*, “Poster: Traffic-Aware Energy Efficiency Optimization in Practical Open RAN 5G Networks,” in *ACM Open-AI RAN Workshop*, 2025, pp. 1–3.
[15] N. Nikaein *et al.*, “OpenAirInterface: A Flexible Platform for 5G Research,” *SIGCOMM Comput. Commun. Rev.*, Oct. 2014.
[16] U. Pawar, B. R. Tamma, and F. A. Antony, “Traffic-Aware Compute Resource Tuning for Energy Efficient Cloud RANs,” in *Proc. 2021 IEEE Global Communications Conference (GLOBECOM)*, 2021.