2023-12-18

Improving Energy Efficiency In Open RAN Through Dynamic CPU Scheduling

Saish Urumkar
Technological University Dublin, d22129017@mytudublin.ie

Byrav Ramamurthy
University of Nebraska-Lincoln, ramamurthy@unl.edu

Sachin Sharma
Technological University Dublin

Improving Energy Efficiency In Open RAN Through Dynamic CPU Scheduling

Saish Urumkar
Technological University Dublin
D22129017@mytudublin.ie
ORCID: 0000-0003-4969-0674

Byrav Ramamurthy
University of Nebraska-Lincoln
ramamurthy@unl.edu
ORCID: 0000-0002-4104-7513

Sachin Sharma
Technological University Dublin
Sachin.Sharma@TUDublin.ie
ORCID: 0000-0002-8358-2258

***Abstract*—Open RAN is a promising cellular technology that is currently undergoing extensive research for future wireless radio access networks. Achieving optimal energy efficiency in Open RAN poses a significant challenge. This paper introduces a CPU scheduling algorithm that specifically targets this challenge by optimizing energy consumption at the base station while maintaining optimal performance levels. With the goal of minimizing energy consumption, the proposed algorithm dynamically adjusts the CPU core states, seamlessly switching between active and sleep modes based on the load conditions. To evaluate the algorithm's effectiveness in terms of energy saving and performance, experimental testing is conducted using the POWDER test-bed located in the United States. The results of the experiments show a reduction of around 5% to 22% in energy consumption in Open RAN through the implementation of the CPU scheduling algorithm.**

***Index Terms*—Open RAN, Energy Efficiency, CPU Scheduling**

I. Introduction

Open Radio Access Network (Open RAN), a rapidly evolving technology, has garnered significant interest in the wireless community. By enabling multi-vendor solutions, supporting openness and promoting interoperability, Open RAN has the potential to revolutionize conventional network designs. Open RAN offers benefits like enhanced network performance, lower operational costs and a diverse ecosystem that empowers network operators to establish flexible and cost-effective networks, fostering adaptability and innovation.

Despite the various benefits that Open RAN offers, the challenge of energy efficiency remains prominent in its deployments [1], [2], as the escalating demand for high-performance networks has led to a substantial increase in base station energy consumption. Efforts to address this challenge have led researchers to explore energy-saving techniques for Open RAN, encompassing dynamic resource allocation, idle power reduction, and machine learning algorithms [3]–[24].

Achieving a balance between energy efficiency and network performance remains a significant challenge. Several papers [4]–[10] suggest altering the power states of network components or reducing active cells, but this approach may compromise network performance and user experience. Sleep mode techniques involve progressive shutdown of network hardware, base station, or cells based on activation and deactivation times [4]–[6]. While sleep mode algorithms for base stations and radio transceivers [4]–[9] have garnered attention, little focus has been given to sleep mode algorithms tailored for CPU cores in Open RAN.

Distinguishing our work from previous studies, such as sleep mode techniques for base stations, cells, and network resources dynamic switching [1]–[25], our research addresses these challenges through a CPU scheduling algorithm designed explicitly for the Open RAN base station (gNodeB) side. Our approach seeks to achieve a balance between energy efficiency and network performance, resulting in a significant energy reduction of around 5% to 22% in our experiments. By dynamically adjusting CPU core states between active and sleep modes based on load, our algorithm optimizes energy consumption during periods of low demand while ensuring efficient resource utilization. Our work establishes a substantial correlation between performance and energy efficiency, offering valuable insights for implementing energy-saving measures in Open RAN systems.

Our research also aligns with the European Union's energy efficiency target of achieving at least 11.7% improvement by 2030 [18]. Our contribution supports ongoing efforts to enhance energy efficiency in Open RAN, aligning with broader goals of sustainable and environmentally-friendly network operations. Researchers evaluating Open RAN performance and energy efficiency have utilized various platforms, including simulation tools like ns-3, mininet, and opennet [19], [20], as well as test-beds featuring commercial and open-source Open RAN implementations [21], [22]. In this paper, we utilize the POWDER test-bed [23], [24] to conduct Open RAN experiments focused on energy efficiency. This paper's contributions are outlined as follows:

1) Deployment of Open RAN Experiment Setup Using POWDER Test-Bed: Our initial setup involves configuring Open RAN on the POWDER test-bed, encompassing essential components such as user equipment (UE), base station (gNodeB) and the Open RAN controller.
2) Proposal of an Energy-Efficient CPU Scheduling Algorithm: We introduce and implement a dynamic CPU scheduling method tailored for gNodeBs. The algorithm performs the balance between energy conservation and operational effectiveness, which is necessary for gN-

odeBs in remote locations as they are susceptible to power variations. Our dynamic CPU scheduling approach efficiently manages power consumption without significantly compromising performance by adjusting CPU core states based on real-time load.

3) Establishing Performance Baseline: We perform benchmarking to establish a reliable performance baseline in our experiments. This phase involves data gathering, focusing on TCP and UDP bandwidth measurements between UE and gNodeB, as well as energy usage and load measurements without implementing the algorithm. This bench-marking provides insights into early performance characteristics.

The remainder of this paper is structured as follows: Section II presents related work. Section III provides an overview and description of the CPU scheduling algorithm. Section IV details the Open RAN emulation configuration used in our experiments. Section V presents results and analysis, highlighting the advantages of the CPU scheduling algorithm. Finally, Section VI concludes the paper by summarizing contributions and suggesting potential directions for further investigation.

II. Related Work

An overview of related work on various proposed methods for energy saving in Radio Access Networks is outlined in the below sections and Table I according to different areas like dynamic activation and deactivation, load balancing and traffic offloading, radio resource management and machine/deep learning.

A. Dynamic Activation And Deactivation

According to network traffic demands, a suggested technique involves dynamic activation and deactivation of base stations or cells [4]–[11]. In paper [4] authors proposed an energy-saving strategy based on green wireless communication to enhance energy efficiency and wireless network effectiveness. They collected data from real base stations and demonstrated that employing techniques like power control, dynamic channel allocation, and sleep mode management led to approximately 18.8% of energy savings when TRX (transceivers) were turned off [4]. In paper [5] authors used simulations to develop an analytical approach for estimating energy savings enhancements at the system level for base stations. The results showed energy savings of 22% for 4G and 17% for 5G under high load conditions of 40% [5]. In the context of green cellular networks, in paper [10] authors introduced methods for dynamic base station switching on/off. Their proposed solutions maintained Quality of Service (QoS) while achieving up to 14% energy savings by selectively activating and deactivating base stations based on traffic demand [10]. In paper [11] authors concentrated on utilizing on/off base stations to operate cellular networks with minimal energy consumption. By strategically powering down base stations during periods of low traffic, they achieved energy savings of up to 4.72% [11].

TABLE I
Related Work on Energy Saving in RAN

Area	Methods (Simulation/Emulation)	Energy Saving%
Dynamic Activation and Deactivation	Real base station for Power control, Dynamic channel allocation, and sleep mode management [4].	Up to 18.8%
	Simulation for base station time- triggered sleep model [5]	Up to 22%
	Simulation for implementing dynamic base station switching-on/off [10]	Up to 14%
	Simulation for implementing adaptive cell zooming for low traffic demand [11]	Up to 4.72%
Load Balancing and Traffic Offloading	Simulation for introducing Pico cells as low-powered heterogeneous network [12]	Up to 9%
	Simulation for traffic load balancing framework [13]	Up to 17%
Radio Resource Management	Simulation for multi-user adaptive power and resource allocation algorithm [14]	Over 77%
Machine/ Deep Learning Algorithms	Simulation to implement linear programming based Energy-efficient power control approach in Open RAN [15]	Up to 40%
	Simulation to implement Multi-Agent Team Learning in virtual Open RAN [16]	Up to 59.04%
	Simulation to implement Traffic prediction method based on recurrent and sequential network [17]	Up to 12.2%

B. Load Balancing And Traffic Offloading

By optimizing resource allocation and evenly distributing traffic among base stations, load balancing solutions aim to reduce excessive energy consumption [12], [13]. In paper [12] authors achieved energy savings by incorporating pico cells as low-power nodes in a heterogeneous network. Their findings indicated approximately 9% power savings for concentrated base stations during busy hours, with minimal average daily power savings of about 1% [12]. The authors in paper [13] proposed a traffic load balancing framework called vGALA for hybrid energy-powered software radio access networks (SoftRAN). Simulations demonstrated that their algorithm maximized energy utilization, reduced network congestion, and led to energy savings of up to 17% [13].

C. Radio Resource Management Techniques

Advanced techniques like intelligent power regulation and interference management contribute to energy efficiency improvements in radio resource management [14]. In paper [14] authors introduced a green radio solution using a multi-user adaptive power and resource allocation algorithm for base stations. Their link adaptation, multi-user diversity and allocation of spare spectrum, through simulation demonstrated over 77% energy savings compared to non-energy aware schemes [14].

D. Machine/Deep Learning Algorithms

Various machine learning and deep learning algorithms have been proposed to optimize energy efficiency in RAN

systems. The authors in paper [15] introduced a joint optimization solution using mixed-integer linear programming to enhance energy efficiency in Open RAN, achieving up to 40% energy savings compared to the baseline solution [15]. In another paper [16] authors presented a Multi-Agent Team Learning (MATL) approach, demonstrating that Team Multi-Agent Deep Reinforcement Learning (TMADRL) outperforms other models, yielding energy gains of 29.7% to 59.04% [16]. The authors in paper [17] focused on traffic prediction using recurrent neural networks, revealing a 12.2% improvement in energy efficiency over standard methods [17].

While the methods discussed in references [2] to [17] have shown positive outcomes, there remain unexplored problems in Open RAN's energy efficiency. Balancing energy efficiency with network performance is a challenge. Existing research has focused on sleep modes for radios and base stations, neglecting CPU cores used in Open RAN. Unlike previous approaches [1] to [17], our proposed CPU scheduling algorithm at the gNodeB effectively conserves energy while maintaining performance, as demonstrated across various link capacities.

III. Proposed CPU Scheduling Algorithm

The proposed algorithm is divided in to two parts: The explanation of objective function and proposed CPU scheduling algorithm.

A. Objective Function

$$\text{minimize: } P(x)\text{total} = \sum_{i,n=1}^{Cn,N} P(U(c)) \tag{1}$$

Eq. 1 represents the objective function for minimizing the total power consumption denoted by $P(x)$total. The objective is to find the optimal solution that minimizes the total power consumption, taking into account the power consumption function $P(U(c))$ for each CPU core c. The summation notation $\sum_{i,n=1}^{Cn,N}$ indicates that the summation is performed over two indices: i and n. The variables i and n represent the indices for the CPU core and the tasks or processes, respectively. The range of i and n extends from 1 to Cn and from 1 to N respectively. Here, Cn represents the total number of CPU cores, while N represents the total number of tasks or processes. This implies that the objective function considers the power consumption of each combination of CPU cores and tasks or processes. $P(U(c))$ represents the power consumption function for a given CPU utilization $U(c)$ of the CPU core c.

B. CPU Scheduling Algorithm

The proposed CPU scheduling algorithm efficiently manages CPU cores by adapting to workload changes. It adjusts the state of a number of active cores based on CPU load, saving power. When the load drops below a threshold, it changes the state of CPU core to sleep mode, conserving energy. Core state changes to active when the load exceeds the threshold, ensuring resource availability. The algorithm optimizes CPU core state swiftly, depending on varying loads. The goal of our proposed algorithm is to switch between sleep and active state depending on load conditions to save power.

Algorithm 1 Dynamic CPU sleep scheduling

```
N: total cores, ns: time, active_cores: number of active cores
cpu_load_threshold: threshold, current_load: current load, gn_pid: Gnodeb PID
Observe and Set gn_pid
while system is running do
    Monitor peak load for ns
    if load < threshold and active_cores > 1 then
        Sleep one core, Decrement active_cores
    end if
    if load > threshold and sleeping cores exist then
        Wake one core, Increment active_cores
    end if
    if no sleeping cores and active_cores = N then
        Wait for ns before monitoring again
    end if
    if no active cores and sleeping cores exist then
        Wake one core
    end if
    if active cores and high load then
        Adjust scheduling
        Implement changes based on observed load
    end if
end while
```

Algorithm 1 begins by setting an initial CPU affinity for the target process, specified by the GNB process ID indicated as '*gn_pid*' in the algorithm. CPU affinity allows us to assign one or more processes to a particular CPU core so that processes can only be executed from that core. The CPU affinity is initially set on all available CPU cores. A load threshold indicated as '*cpu_load_threshold*' value is assigned to determine the threshold limit of load for the algorithm to trigger based on load conditions. The algorithm continuously monitors the CPU load in '*current_load*'. If the CPU load is indicated as '*current_load*' falls below the specified '*cpu_load_threshold*' and there is more than one active CPU core, the algorithm places one of the CPU cores into a sleep state. The CPU affinity is then adjusted to exclude the sleeping core, resulting in decreased power consumption. Conversely, if the CPU load exceeds the threshold based on '*cpu_load_threshold*' and there are available CPU cores, the algorithm activates an additional CPU core and restores the performance mode. The new core is added to the CPU affinity, allowing the process to utilize more computing resources. Throughout the execution, the algorithm prints the CPU load percentage, the adjusted CPU affinity for the GNB process, and the state of individual CPU cores.

IV. Open RAN Emulation Setup

A. About The Powder Platform for OPEN RAN Use Case

POWDER (Platform for Open Wireless Data-driven Experimental Research) is a cutting-edge wireless research platform that allows researchers to conduct experiments in a simulated and controlled environment [23], [24], [25]. Emulation of Open RAN (Radio Access Network) is one of POWDER's primary use cases, allowing researchers to assess and validate the performance of Open RAN designs and protocols [24], [25]. Open RAN emulation enables researchers to evaluate and compare the performance, scalability, and efficiency of Open RAN systems under various scenarios and conditions. In POWDER, emulation scenarios involve configuring virtualized network services such as base stations, access points, and core network components, as well as modeling realistic traffic patterns and network conditions.

B. Emulation Scenario for Open RAN setup in POWDER

The d430 hardware [27] which is powered by 64-bit Intel Xeon processors having thermal design power of 85 watts is used in the POWDER experimental setup. The processor has 2 CPU sockets, 32 cores with hyper-threading technology, and 64 GB of RAM. In the emulation, Docker containers are used to create virtual instances of User Equipment, gNodeB, and Core Network. The OpenAirInterface (OAI) framework repository [28] is implemented to simulate GNB, UE, and CN, adhering to Open RAN specifications. The network function containers for the Access And Mobility Management Function (AMF), User Plane Function (UPF), Session Management Function (SMF), and Network Repository Function (NRF) are deployed within the Core Network. The experiment starts by initializing Docker containers for gNodeB, UEs and Core Network. AMF logs monitor UE and GNB registration. The GNodeB begins in standalone mode using RF simulation. Separate namespaces assign unique IP addresses to three UEs. The nr-soft modem software which is software-defined radio (SDR), emulates GNodeB functions. For UEs, nr-uesoftmodem software is employed, operating within the frequency range 1 (FR1) with sub 6GHz frequencies and *numerology 1* configuration. In 5G NR, the sub-carrier spacing for *numerology 1* is 30 kHz. Also, 106 resource blocks are used in configuring UE. Below Figure 1 is a representation of the Open RAN emulation setup done in POWDER.

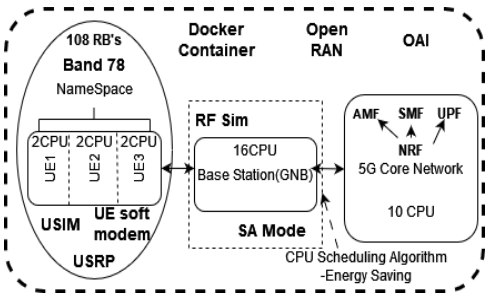

Fig. 1. Open RAN Emulation Setup

C. Data Collection Methodology

Our data collection approach comprises two distinct scenarios, each testing the CPU scheduling algorithm's impact:

1) Scenario 1: Data Collection without CPU Scheduling Algorithm: In this initial scenario, we establish a baseline by conducting data collection without implementing a scheduling algorithm. Using a *bash* script and the *iperf3* utility, we facilitate simultaneous data transmissions between three UE devices and a gNodeB. Our collection encompasses TCP and UDP protocols across link capacities ranging from 500 to 6000Mbps. Additionally, we employ the *turbostat* program to monitor CPU core power consumption, tracking package watts, temperature, and load percentage.

2) Scenario 2: Data Collection with CPU Scheduling Algorithm: Moving to the next phase, we introduce the CPU scheduling algorithm into the data collection process. While the core methodology remains consistent with the previous scenario, the algorithm's presence adds another layer of complexity. Again, three UE devices and the gNodeB engage in simultaneous *iperf3*-based interactions, spanning link capacities from 500 to 6000Mbps and involving both TCP and UDP protocols. Our monitoring of power consumption metrics remains unchanged, with over 100 data points recorded at one-second intervals.

V. Results

Using the data collected in the previous section we demonstrate results using 9 plots as follows:

A. Comparison of Energy Savings

The algorithm's effectiveness is highlighted with a significant reduction of energy consumption of up to 20.1% for a 6000 Mbps link capacity (see Figure 2).

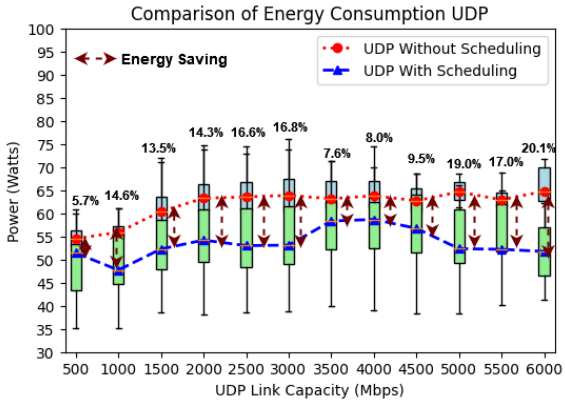

Fig. 2. Energy Savings in UDP

There is a minor decrease in savings observed for link capacities ranging from 3500 to 4500 Mbps. This can be attributed to the challenges of attaining higher UDP bandwidth within the 3000 to 4500 Mbps range. This range is significant because it requires additional computational resources to maintain the achieved UDP bandwidth levels, which tend to fluctuate between 3500 Mbps and 4500 Mbps as evident in

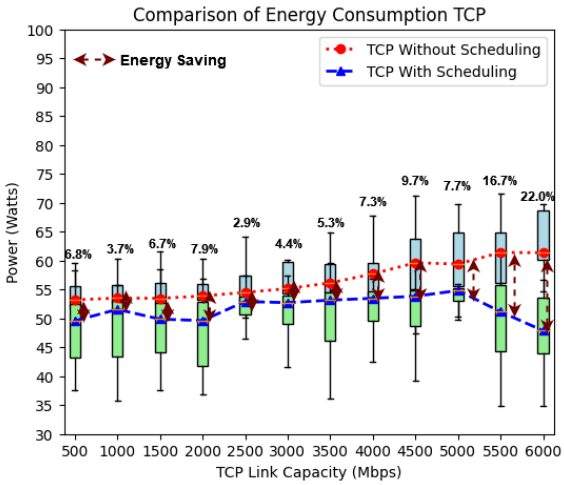

Fig. 3. Energy Savings in TCP

Figure 8. This particular range holds importance due to the experiment's utilization of band 78. Regarding TCP, energy savings range from 6.8% to 22% for link capacities of 500 Mbps to 6000 Mbps (see Figure 3). A minor decline in savings is observed for link capacities between 2500 and 4000 Mbps due to band 78 frequency.

B. Load vs Link Capacity Comparison

The scheduling algorithm slightly reduces the observed load compared to without the scheduling algorithm. By dynami-

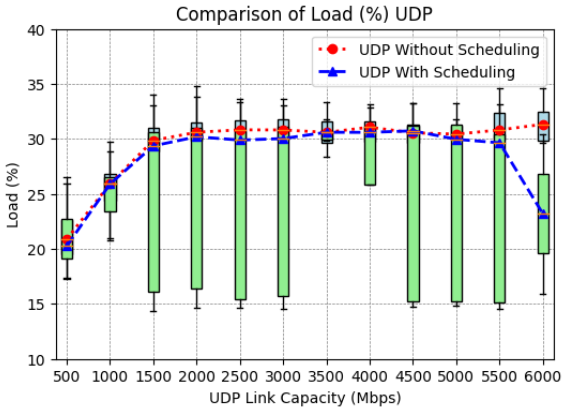

Fig. 4. UDP Load

cally managing CPU core states, the algorithm decreases the cumulative CPU core load percentage (see Figure 4).

TCP shows a small load percent change (see Figure 5). The load increases linearly from 15% to a peak of 27%, followed by a slight drop to about 24% at 6000 Mbps link capacity. By dynamically managing CPU core states, the algorithm decreases the cumulative CPU core load percentage (see Figure 4). TCP shows a small load percent change (see Figure 5). The load increases linearly from 15% to a peak of 27%, followed by a slight drop to about 24% at 6000 Mbps link capacity.

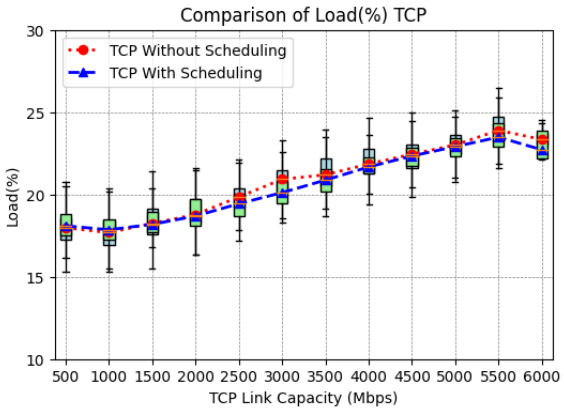

Fig. 5. TCP Load

C. Load vs Power Comparison

Figure 6 shows a clear relationship i.e. as load increases, power consumption rises. From Figure 6, we can see that

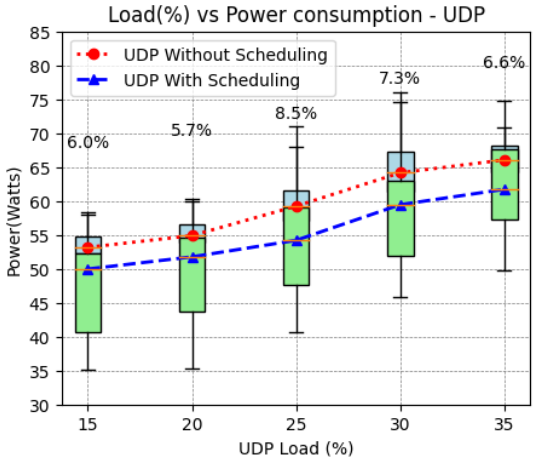

Fig. 6. UDP Load vs Power

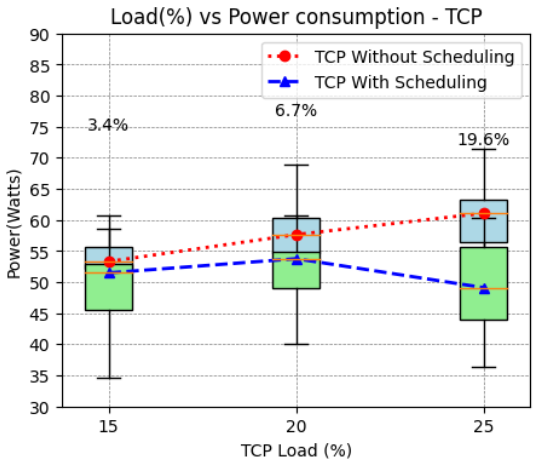

Fig. 7. TCP Load vs Power

using a CPU scheduling algorithm reduces power usage com-

pared to without CPU scheduling algorithm, even at high loads. Impressively, at 35% load, the algorithm saves about 6.6% energy. TCP's load, shown in Figure 7, is lower than UDP load. The power consumption also decreases with higher load percentages. Unlike the UDP plot, TCP plots are linear, also the load is less (see Figures 4 and 5). In TCP, our algorithm achieves significant energy savings of around 19.6% at a maximum load of 25%.

D. Performance Comparison

In Figure 8, UDP achieved bandwidth with the scheduling algorithm is lower than without the scheduling algorithm, as link capacity increases from 3000 to 6000 Mbps. The highest

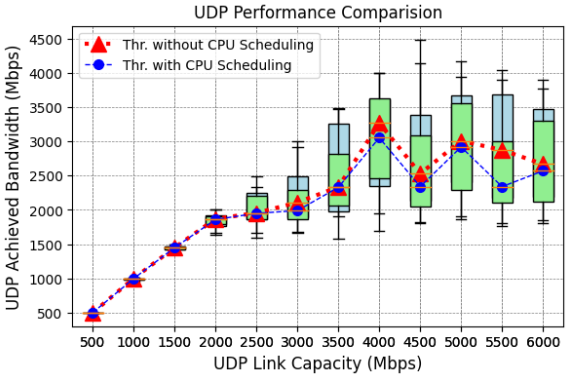

Fig. 8. UDP Performance

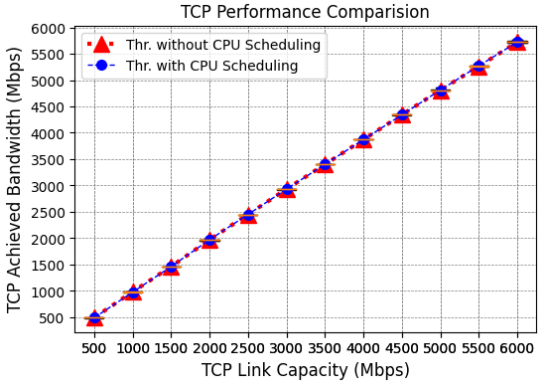

Fig. 9. TCP Performance

achieved bandwidth is around 3600 Mbps at 4000 Mbps link capacity. Beyond 3000 Mbps, stable higher bandwidth isn't achieved. Figure 9 shows TCP tests with negligible differences with or without the algorithm. The attained bandwidth closely matches the set link capacity. This stability could be due to the controlled environment with stationary UEs and no external disruptions.

VI. Conclusion

In this paper, we proposed a dynamic CPU sleep scheduling algorithm to potentially save power. In our experimental setup we made use of the renowned POWDER test-bed situated in USA. With the POWDER test-bed we were able to emulate real world scenarios and also collect results in near real time. Through the implementation of proposed dynamic sleep scheduling algorithm we were able to achieve energy savings up to 22%. Also, even at high loads our proposed algorithm achieved energy savings up to 19.6%. The effectiveness of the algorithm would be more pronounced in a real Open RAN environment for example, a setup in campus where more traffic would be generated. As part of future work, we would work on evaluating the effectiveness of the proposed CPU scheduling algorithm on various separate hardware nodes, more UEs, and more wireless technologies for the 6G era and beyond.

VII. Acknowledgement

Prof. Ramamurthy was supported, in part, by US National Science Foundation under Grant No. CNS-1817105.

References

[1] N. Aryal et al., "Open Radio Access Network challenges for Next Generation Mobile Network," (ICIN), France, 2023.
[2] M. Peng et al., "Heterogeneous cloud radio access networks:a new perspective for enhancing spectral and energy efficiencies,"Wicom 2014.
[3] Y. Li et al., "Energy-efficient femtocell networks:challenges and opportunities,"Wicom, 2013.
[4] Madhu Sudan Dahal et al., Energy saving technique and measurement in green wireless communication, Energy, 2018.
[5] P. Lähdekorpi et al., "Energy efficiency of 5G mobile networks with base station sleep modes," 2017 IEEE(CSCN), Finland, 2017.
[6] T. Kang, X. Sun and T. Zhang, Base station switching based dynamic energy saving algorithm for cellular networks, IEEE, 2012.
[7] Y. Zhu et al., QoS-aware user association based on cell zooming for energy efficiency in cellular networks, in: PIMRC, IEEE, 2013.
[8] C. Meng et al., A low complex energy saving access algorithm based on base station sleep mode, in: ICCC, IEEE, 2013.
[9] J. Wu et al., "Energy-Efficient Base-Stations Sleep-Mode Techniques in Green Cellular Networks: A Survey," in IEEE CSAT, 2015.
[10] E. Oh et al., "Dynamic Base Station Switching-On/Off Strategies for Green Cellular Networks," in IEEE TWC, May 2013.
[11] Choi Y-H et al. Energy Efficient Operation of Cellular Network Using On/Off Base Stations. IJDSN, 2015.
[12] M.W. Arshad et al., Energy efficiency gains through traffic offloading and traffic expansion in joint macro pico deployment, IEEE, 2012.
[13] T. Han et al., "A Traffic Load Balancing Framework for Software Defined RAN Powered by Hybrid Energy Sources," ACM, 2016.
[14] C. Han etal., "Adaptive Power and Resource Allocation Strategies for Green Radio," IEEE GLOBECOM 2011, USA.
[15] T. Pamuklu et al., "Energy-Efficient and Delay-Guaranteed Joint Resource Allocation and DU Selection in O-RAN,"IEEE(5GWF),2021.
[16] P. E. Iturria-Rivera et al., "Multi-Agent Team Learning in Virtualized Open Radio Access Networks (O-RAN),"Sensors, Jul.2022.
[17] F. Rau et al., "A Novel Traffic Prediction Method Using Machine Learning for Energy Efficiency in Service Provider Networks,"Sensors,2023
[18] "Energy Efficiency Targets." Available online: http://surl.li/kavap.
[19] S. Persia et al., "Ns3 and Mininet Codes to Investigate Complete 5G Networks," 2021 AEIT, Italy, 2021.
[20] M. -C. Chan et al., "OpenNet: A simulator for software-defined wireless local area network," 2014 IEEE (WCNC), Turkey, 2014.
[21] M.Vilakazi et al.,"Evaluating an Evolving OAI Testbed",(ICACCS21).
[22] J. X. Salvat et al., "Open Radio Access Networks (O-RAN) Experimentation Platform: Design and Datasets," in IEEE CM.
[23] Upadhyaya et al., Prototyping next-generation O-RAN research testbeds with SDRs. arXiv preprint arXiv:2205.13178.
[24] S. Urumkar et al., "Demonstrating Configuration of Software Defined Networking in Real Wireless Testbeds," IEEE (LANMAN), USA, 2022.
[25] David Johnson et al. Open source RAN slicing on POWDER: a top-to-bottom O-RAN use case. (MobiSys '21). NY, USA.
[26] Upadhyaya et al., 2022. Prototyping next-generation O-RAN research testbeds with SDRs. arXiv preprint arXiv:2205.13178.
[27] Emulab. "D430 Specifications" Available online: http://surl.li/kauyj.
[28] Utah.edu. "OAI Repository" Available online: http://surl.li/kauxv.