

Open Distributed Unit (O-DU) Dynamic Core and DVFS Optimisation for Energy–Latency Trade-offs

Godlove Suila Kuaban*, Tadeusz Czachórski*, Piotr Czekalski[§], Onyeka J. Nwobodo[§],
Jeremiah O. Abimbola[§], Tulin Atmaca[†]

*Institute of Theoretical and Applied Informatics, Polish Academy of Sciences,
Bałtycka 5, 44-100 Gliwice, Poland, Emails: {gskuaban, tadek}@iitis.pl

[†]SAMOVAR, Télécom SudParis, Institut Polytechnique de Paris, 91120,
Palaiseau, France, Email: tulin.atmaca@telecom-sudparis.eu

[§]Faculty of Automatic Control, Electronics and Computer Science,
Silesian University of Technology, Akademicka 16, 44-100 Gliwice, Poland,
Email: {jeremiah.abimbola, onwobodo, piotr.czekalski}@polsl.pl

Abstract—Energy efficiency is one of the central design challenges for Open Radio Access Network (O-RAN) Distributed Units (O-DUs), which run baseband processing on general-purpose, multi-core platforms whose idle power consumption is far from negligible. This paper presents an analytical framework that jointly optimises the number of active CPU cores and the processor frequency ratio (DVFS) of an O-DU so as to minimise node power while respecting a per-request latency service-level agreement (SLA). The framework couples (i) a workload-decomposition model with hardware-accelerator offloading, (ii) a CMOS-inspired power model with static, idle and dynamic components, (iii) an exact M/M/c queueing latency model based on the Erlang-C formula, (iv) an exponential-moving-average (EMA) traffic smoother that makes the controller react to trends rather than spikes, and (v) a regularised, hysteresis-stabilised discrete optimisation solved by grid search at each control step. Driven by a diurnal traffic profile, the controller parks unused cores and lowers frequency off-peak. Numerical results for a single O-DU serving 4, 8 and 16 radio units (RUs) show energy savings of up to 63.8% at low load while keeping the mean latency close to the SLA target, together with 66–71% reductions in energy per request. The results quantify a favourable power–latency trade-off and provide an implementable control policy for green O-RAN deployments.

Index Terms—O-RAN, O-DU, energy efficiency, DVFS, dynamic core management, M/M/c queue, Erlang-C, latency SLA, 5G/6G networks, green networking.

I. INTRODUCTION

The continued densification of mobile access networks for 5G and the coming 6G era is delivering higher data rates, lower latency and massive connectivity, but at the cost of a rapidly growing energy footprint of the radio access network (RAN). The Open RAN (O-RAN) paradigm disaggregates the classical base station into an Open Radio Unit (O-RU), an Open Distributed Unit (O-DU) and an Open Centralised Unit (O-CU) connected through open interfaces, enabling multi-vendor, virtualised and software-defined deployments [1], [2]. While radio units dominate the absolute power budget through their power amplifiers [3], the O-DU has emerged

as a significant and highly load-sensitive consumer because it executes Layer-2 (RLC, MAC) and upper-PHY processing on general-purpose, multi-core servers whose static and idle power is substantial and only weakly proportional to the offered load [4], [5].

Although O-RAN offers significant benefits in terms of flexibility, openness, and virtualisation, these advantages do not automatically translate into improved energy efficiency. The introduction of additional processing entities, particularly O-DU servers, can increase overall power consumption unless advanced optimisation mechanisms are employed to fully exploit the energy-saving potential of the O-RAN architecture [6].

Energy modelling of cellular access has a long history. The EARTH project established reference power models for base-station components and showed that, at the radio unit, the power amplifier dominates and that load proportionality is poor at low utilisation [3]. The broader observation that general-purpose servers are far from energy-proportional — drawing a large fraction of peak power even when idle — motivates active power management at the compute tier [4]. In the O-RAN context, the disaggregation of the base station into O-RU, O-DU and O-CU, and the choice of functional split, strongly affect where processing energy is spent and how it scales [1], [2], [7]; RAN-as-a-service studies further quantify the efficiency benefits of pooling baseband processing in cloud infrastructure [5].

A second line of work studies processor-level energy management. Classical speed-scaling and DVFS results exploit the convex (typically cubic) dependence of CMOS dynamic power on frequency to trade performance for energy [8], [9], and queueing-theoretic analyses characterise the optimal speed under load, balancing energy against sojourn time or latency in processor-sharing and multi-server systems [10], [11]. Most existing O-RAN power studies, however, either assume linear scaling of O-DU power with load or treat core management and DVFS in isolation, and they rarely couple the power model to an explicit queueing latency constraint. Our contribution is to *jointly* optimise active-core count and DVFS frequency

This work has been part of Celtic-Next project RAI-6Green: Robust and AI Native 6G for Green Networks with project-id: C2023/1-9 funded by the European Commission.

under an exact M/M/c latency SLA, with traffic smoothing, transition costs and hysteresis, so that the resulting policy is both globally optimal over the small discrete action space at each step and stable enough to deploy. The individual ingredients—DVFS, core parking, queueing analysis and EMA smoothing—are well established; the novelty here lies in their tight coupling. An *exact*, closed-form Erlang-C latency SLA is embedded directly in a joint core-and-DVFS decision that is solved to global optimality at every control step and executed as a standard *rApp/xApp*, in contrast to prior O-RAN power studies that assume linear power scaling or treat core management and DVFS in isolation.

A. Main contribution

A core inefficiency of such platforms is that, when provisioned for peak capacity, a large fraction of their cores remain powered and clocked at the nominal frequency even when traffic is light. Two well-known hardware mechanisms can recover this energy: *core parking* (deactivating idle cores) and *dynamic voltage and frequency scaling* (DVFS), which exploits the cubic dependence of CMOS dynamic power on clock frequency [8], [9]. The difficulty is that both mechanisms reduce the available service rate and therefore inflate processing latency, which must remain within strict O-DU service-level agreements (SLAs). The right operating point is thus the solution of an energy–latency trade-off that depends on the instantaneous traffic, the queueing behaviour of the processing pipeline and the cost of switching hardware states [10]. It is essential to develop an analytical framework that is both simple and practically applicable.

This paper develops and evaluates a compact, physically grounded framework that resolves this trade-off online. Its contributions are:

- a workload-decomposition model that maps aggregate RU cycle demand onto CPU cycle demand after accelerator offloading and resource pooling;
- a CMOS-inspired node power model and a single-core throughput/utilisation model that together capture how active-core count and frequency shape both energy and capacity;
- an exact M/M/c latency model (Erlang-C) that yields the mean sojourn time (latency) as a closed-form function of the control variables, with an EMA traffic smoother for stable decisions;
- a regularised discrete optimisation with transition penalties and a hysteresis dead-band, solved by grid search, that produces a stable, implementable control policy; and
- a numerical evaluation over a five-day diurnal profile reporting the achieved energy savings, the power–latency product and the energy-per-request efficiency.

II. SYSTEM MODEL

We consider an O-RAN network in which a single O-DU node serves K radio units over the fronthaul network and forwards aggregated traffic to the O-CU via the midhaul network, following the 3GPP Option 7-2 functional split, in

which only the lower-PHY remains at the O-RU [2], [7]. An abstract representation of the O-DU node consists of a multi-core CPU complex, an optional hardware accelerator (ASIC, FPGA or GPU) for highly parallel baseband functions, and a platform and cooling subsystem. Fig. 1 shows the three-layer architecture of an O-RAN and the position of the O-DU.

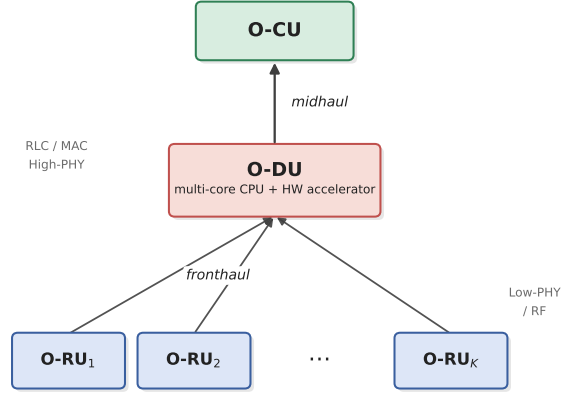


Fig. 1. Three-layer O-RAN architecture. The O-DU aggregates K radio units over the fronthaul and runs Layer-2 and upper-PHY baseband processing on a multi-core platform with an optional accelerator; this paper optimises the O-DU active-core count and DVFS frequency.

The controller chooses, at each control step, the number of active cores N_{act} and the DVFS frequency ratio f ,

$$N_{\text{act}} \in \{N_{\min}, \dots, N_{\max}\}, \quad f \in \mathcal{F} = \{f_1, \dots, f_L\}, \quad (1)$$

where $\mathcal{F}$ is the discrete set of admissible P-states exposed by real hardware. The objective is to minimise node power subject to a latency SLA,

$$\min P_{\text{node}} \quad \text{s.t.} \quad W \leq W_{\text{target}}, \quad (2)$$

with W the mean per-request sojourn time and W_{target} the SLA target. Because the control interval Δt is fixed, minimising the instantaneous node power P_{node} is equivalent to minimising the per-step energy $E = P_{\text{node}} \Delta t$; the cumulative energy is $E_{\text{cum}} = \sum_t P_{\text{node}}(t) \Delta t$, so the energy-saving objective and the power objective coincide. The hard SLA constraint in (2) is enforced online through the soft, one-sided penalty of (14).

III. COMPUTE WORKLOAD, POWER CONSUMPTION, AND PERFORMANCE MODELLING

A. Workload, Traffic and Latency Models

Each radio unit $k \in \{1, \dots, K\}$ generates a cycle demand $C_{\text{RU}}(k)$ (in Mcycles/s). The aggregate raw demand at full traffic is

$$C_{\text{raw}} = \psi \sum_{k=1}^K C_{\text{RU}}(k), \quad (3)$$

where $\psi > 0$ is a workload scaling factor that captures modulation order, coding rate and scheduling density not

modelled per RU. A fraction $\gamma_{\text{acc}} \in [0, 1]$ of the work is offloaded to the accelerator, and $S_{\text{pool}} \in (0, 1]$ is the pooling gain of pooling compute demand from several RUs to the CPUs of the O-DU [12]. At a normalised traffic level $\tau \in [0, 1]$ the CPU cycle demand is therefore

$$C_{\text{cpu}}(\tau) = \psi S_{\text{pool}} (1 - \gamma_{\text{acc}}) \tau \sum_{k=1}^K C_{\text{RU}}(k), \quad (4)$$

which is linear both in traffic τ and in RU aggregation K .

The base diurnal profile is a superposition of an afternoon business peak and an evening residential peak over a constant base-load floor [13],

$$d(t) = b + A_1 e^{-\frac{(t-\mu_1)^2}{2\sigma_1^2}} + A_2 e^{-\frac{(t-\mu_2)^2}{2\sigma_2^2}}, \quad (5)$$

with $b=0.12$, $A_1=0.95$, $\mu_1=14$ h, $\sigma_1=3.2$ h, and $A_2=0.65$, $\mu_2=20$ h, $\sigma_2=2.1$ h. It is normalised to $d_{\text{norm}}(t) = d(t)/\max_t d(t) \in (0, 1]$, and a stochastic realisation is obtained as $\tau(t) = \text{clip}(d_{\text{norm}}(t) + \epsilon(t))$ with zero-mean noise $\epsilon(t)$. To prevent the controller from chasing transient spikes (“flapping”), the load is smoothed by an exponential moving average [14],

$$\tau_{\text{ema}}(t) = \alpha \tau(t) + (1 - \alpha) \tau_{\text{ema}}(t - \Delta t), \quad (6)$$

and all control decisions are taken on τ_{ema} .

B. Latency Model: M/M/c with Erlang-C

Requests are assumed to arrive as a Poisson process of rate λ , set by the CPU cycle demand and the mean per-request cost c_{req} , while each of the M_{ht} simultaneous-multithreading (SMT) threads of an active core acts as a server:

$$\lambda = \frac{C_{\text{cpu}}(\tau)}{c_{\text{req}}}, \quad c = N_{\text{act}} M_{\text{ht}}, \quad \mu = \frac{C_{\text{eff}}(f)}{c_{\text{req}}}, \quad (7)$$

where $C_{\text{eff}}(f)$ is the per-core effective throughput defined in (10). The system is stable when $\rho = \lambda/(c\mu) < 1$. The probability that an arriving request must wait is given by the Erlang-C formula

$$C(c, \rho) = \frac{\frac{a^c}{c!} \frac{1}{1-\rho}}{\sum_{k=0}^{c-1} \frac{a^k}{k!} + \frac{a^c}{c!} \frac{1}{1-\rho}}, \quad a = \frac{\lambda}{\mu}, \quad (8)$$

from which the mean waiting and sojourn times follow [11]:

$$W_q = \frac{C(c, \rho)}{c\mu - \lambda}, \quad W = W_q + \frac{1}{\mu}. \quad (9)$$

As $\rho \rightarrow 0$, $W \rightarrow 1/\mu$ (service time only); as $\rho \rightarrow 1^-$, $W \rightarrow \infty$; and the model reduces to M/M/1 for $c = 1$. M/M/c is a conservative choice for O-DU pipelines, whose service is closer to deterministic (M/D/c) and would yield somewhat lower delays.

C. Throughput and Power Models

At frequency ratio $f \in [f_{\text{min}}, 1]$ the effective throughput per physical core is

$$C_{\text{eff}}(f) = C_{\text{core}} (1 - h) M_{\text{ht}} f, \quad (10)$$

where $(1 - h)$ discounts OS/scheduler overhead (context switches, interrupts) and M_{ht} is the SMT capacity multiplier. With N_{act} active cores the per-core utilisation is

$$u_{\text{core}}(\tau, N_{\text{act}}, f) = \min\left(1, \frac{C_{\text{cpu}}(\tau)}{N_{\text{act}} C_{\text{eff}}(f)}\right), \quad (11)$$

which is capped at the physical limit and decreases as N_{act} or f grows.

Dynamic power follows the CMOS switching law, growing with utilisation and super-linearly with frequency,

$$P_{\text{dyn}}(u_{\text{core}}, f) = P_{\text{core_dyn}} f^{\beta_{\text{freq}}} u_{\text{core}}^{\alpha_{\text{dyn}}}, \quad (12)$$

with $\beta_{\text{freq}} \geq 2$ (typically ≈ 3 under the $P \propto f^3$ rule) and $\alpha_{\text{dyn}} \geq 1$ capturing super-linear effects at high utilisation [8]. The total node power adds the platform, accelerator and per-core idle contributions and accounts for cooling overhead η_{cool} ,

$$P_{\text{node}} = \frac{1}{\eta_{\text{cool}}} \left[P_{\text{plat}} + P_{\text{acc}} + N_{\text{act}} (P_{\text{core_idle}} + P_{\text{dyn}}) \right]. \quad (13)$$

Equation (13) is monotonically increasing in N_{act} , f and u_{core} ; energy efficiency therefore favours fewer active cores and lower frequency, subject to the latency constraint of (9). This is the tension the optimiser must resolve: increasing N_{act} or f raises P_{node} but lowers W , whereas decreasing them saves power at the risk of violating the SLA.

IV. JOINT ENERGY-LATENCY OPTIMISATION

At every control step the controller selects (N_{act}, f) to minimise a single regularised cost that balances energy, latency and stability,

$$J(N, f) = w_e [P_{\text{node}}(\tau_{\text{ema}}, N, f) + P_{\text{trans}}(N)] + w_l \max(0, W(N, f) - W_{\text{target}}) + P_{\text{util}}(N, f), \quad (14)$$

where w_e, w_l weight energy and latency. The latency term is a one-sided penalty that is inactive while the SLA holds, so the optimiser is free to trade latency headroom for power. Equation (14) is thus a soft-constrained (penalty) reformulation of the constrained problem (2): the hard requirement $W \leq W_{\text{target}}$ is represented by the one-sided term, and a sufficiently large w_l drives the optimiser towards feasibility. We set $w_e=1$ so that w_l alone prices latency relative to power. The transition penalty discourages needless hardware-state changes,

$$P_{\text{trans}}(N) = \Delta_N^+ \frac{E_{\text{wake}}}{\Delta t} + \Delta_N^- \frac{E_{\text{sleep}}}{\Delta t}, \quad (15)$$

with Δ_N^+ and Δ_N^- the numbers of cores woken or put to sleep relative to the previous step, and the utilisation penalty

$P_{\text{util}} = \kappa \max(0, u_{\text{core}} - u_{\text{max}})$ keeps cores away from saturation. In the evaluation the transition energies are $E_{\text{wake}}=2$ J and $E_{\text{sleep}}=0.8$ J (a wake-up is priced above a sleep), the weights are $w_e=1$ and $w_l=30$, the utilisation reference is $u_{\text{max}}=0.8$, and the hysteresis band is $\delta_{\text{hyst}}=0.03$; these values are collected in Table I.

A. Grid Search and Hysteresis

The feasible set $\Omega = \{N_{\min}, \dots, N_{\max}\} \times \mathcal{F}$ is small, so the global minimiser

$$(N^*, f^*) = \arg \min_{(N, f) \in \Omega} J(N, f) \quad (16)$$

is found by exhaustive search at every step. To suppress oscillations between near-equal states, a relative hysteresis dead-band is applied: with $J_{\text{opt}} = J(N^*, f^*)$ and $J_{\text{keep}} = J(N_{\text{prev}}, f^*)$, the controller switches only if

$$J_{\text{keep}} - J_{\text{opt}} > \delta_{\text{hyst}} J_{\text{keep}}, \quad (17)$$

and otherwise retains the previous core count N_{prev} . The result is a stable, rate-limited adaptation. Because P_{node} is increasing and W is decreasing in N , the cost is unimodal (discretely quasi-convex) in N for fixed f , which makes the small grid search both exact and inexpensive.

B. Complexity and Implementation

Since the search space contains only a few hundred candidate configurations, exhaustive search is computationally feasible. Because the policy reads only the smoothed load, the current core/frequency state and a handful of static parameters, it maps directly onto an O-RAN *rApp/xApp* running on the RAN Intelligent Controller, with the wake/sleep and P-state transitions issued through standard platform power-management interfaces. The hysteresis dead-band bounds the switching rate, protecting hardware from excessive wear and keeping the amortised transition energy of (14) small. The cost prices the *energy* of wake/sleep and P-state transitions through P_{trans} ; their *latency* (wake-up and voltage/frequency settling times) is not modelled explicitly. The hysteresis band limits the switching rate so that such transients remain infrequent, but explicitly accounting for transition latency is a worthwhile refinement, which we leave to future work.

The per-step and cumulative energies are $E(t) = P_{\text{node}}(t) \Delta t$ and $E_{\text{cum}} = \sum_t E(t)$, and the relative energy saving against a static baseline is $\text{savings} = (E_{\text{base}} - E_{\text{opt}})/E_{\text{base}}$. Two efficiency indicators complete the evaluation. The *power-latency product* (PLP) captures QoS-aware efficiency,

$$\text{PLP}(t) = P(t) T(t) \quad [\text{W} \cdot \text{ms}], \quad (18)$$

penalising operating points that are simultaneously power-hungry and slow, while the *energy per request* (EPR) is the throughput-normalised KPI

$$\text{EPR}(t) = \frac{P(t)}{\lambda(t)} \quad [\text{J/req}]. \quad (19)$$

TABLE I
REPRESENTATIVE MODEL PARAMETERS.

Symbol	Value	Symbol	Value
C_{core}	500 Mc/s	P_{plat}	50 W
C_{RU}	1000–1250 Mc/s	P_{acc}	20 W
c_{req}	0.9 Mc/req	$P_{\text{core_idle}}$	15 W
h	0.10	$P_{\text{core_dyn}}$	40 W
M_{ht}	2	β_{freq}	3.0
γ_{acc}	0.30	α_{dyn}	1.20
α (EMA)	0.35	η_{cool}	0.90
W_{target}	2.5 ms	$\mathcal{F}$	$\{0.40, \dots, 1.00\}$
E_{wake}	2 J	E_{sleep}	0.8 J
w_e	1	w_l	30
δ_{hyst}	0.03	u_{max}	0.8

V. NUMERICAL RESULTS

We evaluate one O-DU box, provisioned for the largest deployment ($N_{\min} = 2$, $N_{\max} = 16$ cores), serving $K \in \{4, 8, 16\}$ RUs over five days at a 10-minute control interval. The *baseline* keeps the whole provisioned core pool powered at $f = 1$; the *dynamic* policy applies (14). Representative parameters are listed in Table I; the evaluation uses $\psi = 1$ and $S_{\text{pool}} = 1$, and the per-core power coefficients, $\beta_{\text{freq}} = 3$ and the cooling overhead reflect commodity server-class hardware.

Fig. 2 shows the node power over five days. Both policies track the diurnal traffic, but the baseline remains high off-peak because its cores stay powered, whereas the dynamic policy collapses to a small active set at night. Integrating the curves yields energy savings of

$$63.77\% (K=4), \quad 60.34\% (K=8), \quad 56.49\% (K=16).$$

Savings fall as the RU count grows: with more aggregated traffic, a larger fraction of the fixed core pool must stay active even off-peak, leaving less to switch off. This is precisely the regime in which statistical multiplexing already keeps utilisation high. This decline in *relative* savings is a direct consequence of the fixed-capacity provisioning—a single 16-core pool sized for the peak—rather than a limitation of the control framework itself: at larger K a greater share of that fixed pool must remain active, so less of it can be parked. The *absolute* efficiency in fact improves with consolidation, as the monotonically falling optimised energy-per-request in Table III shows ($0.145 \rightarrow 0.084 \rightarrow 0.054$ J/req for $K=4, 8, 16$): more aggregated traffic keeps the active cores better utilised, lowering the energy spent per served request even as the parkable fraction shrinks. Relative energy saving and absolute energy efficiency therefore move in opposite directions with K , and should be read together.

The top panel in Fig. 3 shows the controller tracking the diurnal load: the active-core count follows the traffic envelope as a clean staircase, rising to roughly eight cores at the afternoon/evening peaks and collapsing to the two-core floor overnight, while the DVFS ratio moves between its 0.4 minimum off-peak and about 0.55–0.70 at peak. The staircase is smooth rather than chattering, which is the visible signature of the EMA smoother and the hysteresis dead-band suppressing

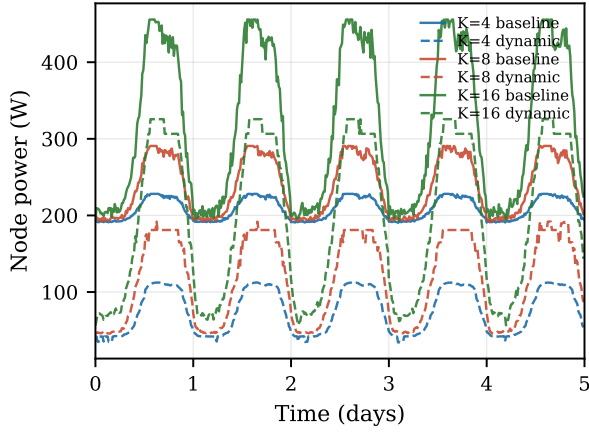


Fig. 2. Node power over five days for $K \in \{4, 8, 16\}$ RUs: baseline (solid) versus dynamic core/DVFS control (dashed). The dynamic policy follows the diurnal load and parks cores off-peak.

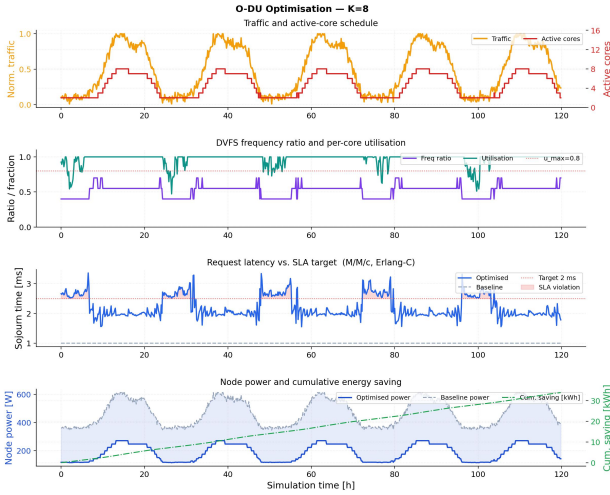


Fig. 3. Four-panel overview ($K = 8$). The active-core schedule (top) tracks the traffic envelope with a clear day/night contrast; the power trace (bottom) shows the optimised node drawing a small fraction of the baseline, with cumulative saving accruing approximately linearly.

flapping despite the noisy load. The second panel explains why this is efficient: per-core utilisation is deliberately held high—close to one, mostly above the $u_{\max} = 0.8$ reference—so the few cores that remain awake are kept productive rather than idling, which is exactly where the CMOS power model rewards consolidation. The dips in utilisation coincide with the ramp transitions, where the controller has briefly added a core ahead of demand. The fourth panel quantifies the payoff: optimised node power swings between roughly 110 W at night and 270 W at peak, against a baseline that never falls below about 350 W, and the cumulative-saving trace climbs nearly linearly to about 34 kWh over the five days—the $\approx 60\%$ energy reduction reported for $K = 8$.

Panel A in Fig. 4 shows the power-latency product improving only moderately (-15.6%): the optimised curve sits below the baseline at all times but the gap is modest, because

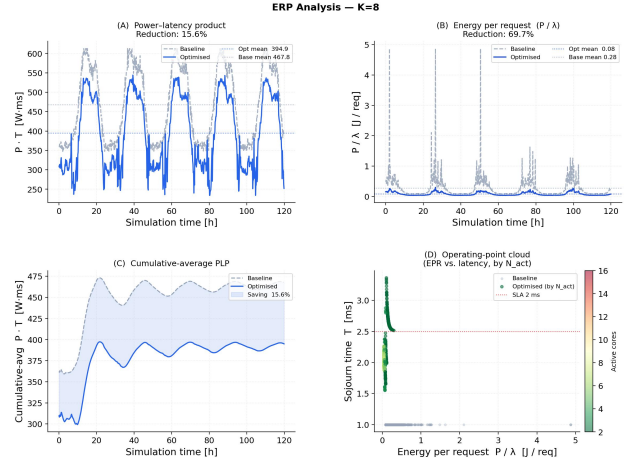


Fig. 4. Energy-response-time product (ERP) analysis ($K = 8$). Panels (A)/(C): power-latency product over time and its cumulative average. Panel (B): energy per request. Panel (D): operating-point cloud coloured by active-core count.

the policy deliberately spends part of its latency headroom—running hotter and slower—to buy the large power cut, so P falls while T rises and the product moves less than either factor alone. Panel B is the strong result (-69.7%): the baseline energy-per-request spikes violently at night, because with the full core pool powered but the request rate collapsing, P/λ blows up at low load; the optimised policy parks cores so that power tracks demand, keeping EPR low and almost flat across the whole cycle. This contrast is the paper’s central efficiency message—energy saving and energy efficiency are different quantities, and EPR, which normalises by useful work, exposes the over-provisioning that raw power does not. Panel C confirms the PLP gap is stable, the cumulative average converging to a steady $\approx 16\%$ separation after the first day’s transient. Panel D is the trade-off frontier made explicit: the baseline points cluster at low latency (≈ 1 ms) but spread across a wide, often high EPR, whereas the optimised points form a low-EPR band whose latency rises as the active-core count falls (the green-to-yellow colouring), i.e. the controller is sliding along an energy-latency frontier, trading sojourn time for efficiency while staying bounded.

Tables II and III summarise the two efficiency indicators. The PLP improves moderately ($14\text{--}20\%$): the policy trades a slight latency increase for a large power reduction, with the largest gains at low load where the baseline is most over-provisioned relative to the offered work. The EPR tells a stronger story, improving by $66\text{--}71\%$, again largest at low load where both the power and the request rate fall, producing a super-linear efficiency improvement. The contrast between the two tables underlines that EPR, which normalises energy by useful work, better reflects true green-network performance.

Three trends emerge. First, the largest energy savings occur at low load, where most of the provisioned pool can be parked. Second, the largest PLP improvement also occurs at low load: there the baseline is most over-provisioned relative to the

TABLE II
POWER-LATENCY PRODUCT (PLP), IN $W \cdot ms$.

K	Baseline	Optimised	Reduction
4	400.0	319.5	20.1%
8	467.8	394.9	15.6%
16	626.4	538.8	14.0%

TABLE III
ENERGY PER REQUEST (EPR), IN J/REQ.

K	Baseline	Optimised	Reduction
4	0.505	0.145	71.4%
8	0.277	0.084	69.7%
16	0.159	0.054	66.1%

offered work, so parking yields a large power cut for only a small latency increase. Third, the EPR improvement is large across the whole operating range (66–71%) and likewise peaks at low load, confirming that the controller raises efficiency everywhere it operates.

The controller keeps the mean per-request sojourn time *bounded and close* to the 2.5 ms target rather than strictly below it. The excess is small—at most ≈ 0.9 ms (worst case ≈ 3.3 ms)—and for $K=8$ the target is met in about 70% of the 10-minute control steps. These overshoots concentrate in two regimes: deep off-peak, where the controller selects the minimum P-state whose bare service time $1/\mu \approx 2.5$ ms already coincides with the target, so that any queueing pushes the mean marginally above it; and brief traffic ramp-ups, before an additional core is brought online. Because the SLA enters (14) as a one-sided *soft* penalty, the policy deliberately rides along the target to harvest energy. A strict bound can be imposed either by increasing the latency weight w_l or by discarding infeasible (N_{act}, f) candidates from Ω , at the cost of a modest reduction in energy savings; likewise, latency-critical DUs with sub-millisecond real-time budgets are accommodated by the same framework, unchanged, with a tighter W_{target} and a higher minimum P-state. This makes explicit the trade-off between energy savings and quality of service.

VI. CONCLUSION

We presented a unified, physically self-consistent framework for O-DU energy-latency optimisation that jointly controls active-core count and DVFS frequency. The framework combines workload decomposition with accelerator offloading, a CMOS-inspired node power model, an exact M/M/c (Erlang-C) latency model, EMA-based traffic smoothing and a regularised, hysteresis-stabilised grid-search controller. Over a five-day diurnal profile it delivers up to 63.8% energy savings and 66–71% reductions in energy per request while keeping the mean latency close to the SLA target (with small, bounded overshoots), and it cleanly separates the notions of energy saving and energy efficiency through the PLP and EPR metrics.

Future work will replace M/M/c with M/G/c or M/D/c service models that better match deterministic baseband pipelines, validate the controller against measured O-DU traces and real P-state transition costs on a hardware testbed, extend the model to heterogeneous CPU/GPU/FPGA platforms, and explore learning-based control policies that anticipate traffic rather than react to it. We also plan to benchmark the policy against alternative O-RAN power-management schemes—rather than the static baseline alone—so that its advantages over existing approaches can be quantified directly. Integrating the O-DU model with O-RU and O-CU power models would also enable end-to-end O-RAN energy optimisation.

Finally, the proposed methodology is sufficiently general to be extended beyond O-RAN systems. Similar optimisation principles can be applied to edge-cloud servers, virtualised network functions, MEC platforms and other multi-core communication systems where processing resources can be dynamically activated or deactivated according to the offered load while satisfying quality-of-service constraints.

REFERENCES

- [1] O-RAN Alliance, “O-ran architecture description,” Technical Specification O-RAN.WG1.OAD, O-RAN Working Group 1, 2023.
- [2] 3GPP, “Study on new radio access technology: Radio access architecture and interfaces,” Technical Report TR 38.801, Rel-14, 3rd Generation Partnership Project (3GPP), 2017.
- [3] G. Auer, V. Giannini, C. Desset, I. Gódor, P. Skillermark, M. Olsson, M. A. Imran, D. Sabella, M. J. Gonzalez, O. Blume, and A. Fehske, “How much energy is needed to run a wireless network?,” *IEEE Wireless Communications*, vol. 18, no. 5, pp. 40–49, 2011.
- [4] L. A. Barroso and U. Hözl, “The case for energy-proportional computing,” *IEEE Computer*, vol. 40, no. 12, pp. 33–37, 2007.
- [5] D. Sabella, A. de Domenico, E. Katranaras, M. A. Imran, M. di Girolamo, U. Salim, M. Lalam, K. Samdanis, and A. Maeder, “Energy efficiency benefits of RAN-as-a-Service concept for a cloud-based 5G mobile network infrastructure,” *IEEE Access*, vol. 2, pp. 1586–1597, 2014.
- [6] Y. Al-Karawi, H. Al-Raweshidy, and R. Nilavalan, “Power consumption evaluation of next generation open radio access network,” in *2024 IEEE International Conference on Consumer Electronics (ICCE)*, pp. 1–6, IEEE, 2024.
- [7] L. M. P. Larsen, A. Checko, and H. L. Christiansen, “A survey of the functional splits proposed for 5G mobile crosshaul networks,” *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 146–172, 2019.
- [8] T. D. Burd and R. W. Brodersen, “Energy efficient CMOS microprocessor design,” in *Proc. Hawaii Int. Conf. System Sciences (HICSS)*, pp. 288–297, 1995.
- [9] M. Weiser, B. Welch, A. Demers, and S. Shenker, “Scheduling for reduced CPU energy,” in *Proc. USENIX Symp. Operating Systems Design and Implementation (OSDI)*, pp. 13–23, 1994.
- [10] A. Wierman, L. L. H. Andrew, and A. Tang, “Power-aware speed scaling in processor sharing systems,” in *Proc. IEEE INFOCOM*, pp. 2007–2015, 2009.
- [11] L. Kleinrock, *Queueing Systems, Volume 1: Theory*. Wiley-Interscience, 1975.
- [12] G. S. Kuaban, T. Czachórski, T. Atmaca, and P. Czekalski, “Analytical power efficiency models for open radio access network (o-ran),” in *2025 33rd International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS)*, pp. 1–8, IEEE, 2025.
- [13] G. S. Kuaban, T. Czachórski, T. Atmaca, and P. Czekalski, “Power models for o-ran o-du planning and optimisation,” in *IEEE Consumer Communications and Networking Conference (CCNC)*, IEEE, 2026.
- [14] E. Pinheiro, R. Bianchini, E. V. Carrera, and T. Heath, “Dynamic cluster reconfiguration for power and performance,” in *Compilers and operating systems for low power*, pp. 75–93, Springer, 2003.