

Energy-Efficient Parallel Computing for Edge Devices: Techniques, Challenges, and Emerging Research Directions

Nidhi Chahal^{1,*}, Samar Singh², Ritik³

Abstract

Edge computing brings computation closer to data sources, which supports low-latency applications but places substantial demands on energy-constrained and heterogeneous devices. Parallel execution across multicore CPUs, GPUs, FPGAs, and other accelerators can improve responsiveness, but it also complicates scheduling, power management, and resource allocation. This review examines six closely related technique families: energy-aware task scheduling, dynamic voltage and frequency scaling (DVFS), task offloading and resource allocation, artificial-intelligence-based optimization, hardware acceleration, and mathematical optimization. Representative studies show that energy efficiency is obtained through different mechanisms: battery-aware placement and scheduling, coordinated control of voltage and frequency, selective migration of computation, learned policies for dynamic environments, specialized data paths, and formal constrained optimization. Reported improvements are promising but are tied to particular devices, workloads, baselines, and measurement methods; therefore, they cannot be treated as directly comparable. The review also identifies recurring limitations, including limited hardware-level validation, weak cross-platform generalization of learning-based methods, inconsistent benchmarks, and fragmented optimization across system layers. Potential research directions include joint scheduling-DVFS-offloading control, hardware-software co-design, transfer learning, sparse-aware execution, federated optimization, energy-aware accelerators, and renewable-aware edge infrastructure. The central conclusion is that practical edge efficiency requires integrated policies that balance energy, latency, reliability, privacy, and implementation overhead rather than optimizing any single metric in isolation.

Keywords: Edge computing, parallel computing, energy efficiency, heterogeneous architectures, task scheduling, DVFS, task offloading, hardware acceleration

INTRODUCTION

*Author For Correspondence

Nidhi Chahal
E-mail: Nidhi.ece@cgce.edu.in

¹Assistant Professor, Department of Electronics and Communication Engineering, Chandigarh Engineering College-CGC, Landran, Mohali, Chandigarh, India

^{2,3}Student, Department of Electronics and Communication Engineering, Chandigarh Engineering College-CGC, Landran, Mohali, Chandigarh, India

Received Date: September 10, 2026
Accepted Date: September 16, 2026
Published Date: September 20, 2026

Citation: Nidhi Chahal, Samar Singh, Ritik. Energy-Efficient Parallel Computing for Edge Devices: Techniques, Challenges, and Emerging Research Directions. Recent Trends in Parallel Computing. 2026; 13(2): 35–41p.

Edge computing places computation and storage near data-generating devices instead of relying exclusively on remote cloud data centers. This architecture is useful for applications such as industrial monitoring, intelligent transportation, healthcare sensing, and video analytics because local or nearby processing can reduce the communication distance and response time. At the same time, edge nodes are often constrained by battery capacity, thermal limits, memory, and intermittent connectivity. These constraints make energy consumption a system-level concern rather than a hardware-design concern.

Parallel computing is important because edge workloads increasingly include computer-vision

pipelines, signal processing, sensor fusion, and machine-learning inference. These workloads can be divided across CPU cores or mapped to GPUs, FPGAs, and neural accelerators. Heterogeneity improves the range of supported workloads, but it makes resource management difficult: processors have different performance and power characteristics, data movement can dominate computation, and a decision that is effective for one workload may be inefficient for another workload. Therefore, scheduling, DVFS, offloading, and accelerator selection must be considered together.

This review synthesizes the recent literature on energy-efficient parallel computing for edge devices. It focuses on what each technique does, why it is useful, what representative studies report, and which limitations remain. The paper is organized as follows. Section II introduces technical background. Section III summarizes related work. Section IV discusses six technique families. Section V compares representative studies. Sections VI–VIII cover challenges, emerging directions, and research gaps. Section IX concludes the review.

BACKGROUND

An edge system may contain end devices, single-board computers, edge servers, and cloud computing resources. Parallel tasks are commonly represented as independent jobs or directed acyclic graphs (DAGs), in which nodes represent computations and edges represent dependencies. A scheduler maps these tasks to processors while considering the deadlines, data-transfer costs, processor availability, and energy. In heterogeneous systems, the fastest processor is not always the best choice; an efficiency core or accelerator may consume less energy for a slightly longer execution time.

A useful first-order model for CMOS dynamic power is represented as:

$$P_{\text{dynamic}} \approx \alpha C V^2 f \quad (1)$$

where α is switching activity, C is effective capacitance, V is supply voltage, and f is clock frequency. Therefore, reducing the voltage and frequency can reduce the dynamic power, with the voltage often having a particularly strong effect because of the approximate V^2 dependence. Static or leakage power remains even when switching activity is low, and its importance depends on technology and operating conditions. Energy is the integral of the total power over time; lowering the frequency may reduce the instantaneous power but can increase the execution time and leakage energy. Therefore, DVFS must balance dynamic power, leakage, latency, and deadline satisfaction rather than assuming that lower frequency is always more efficient.

Offloading adds communication energy and delay to the computation energy decision. Local execution may be preferable for small tasks or poor channels, whereas nearby edge execution can be useful for computation-intensive tasks when bandwidth and server capacity are available. Accelerators reduce instruction and data movement overhead for suitable kernels; however, they introduce programming, memory, and portability costs. These interactions motivate cross-layer optimization.

RELATED WORK

Recent research covers both practical schedulers and model-based optimization. Llorens-Carrodegua *et al.* proposed SOCCS, an energy-friendly scheduler for Kubernetes-managed single-board-computer clusters. Its placement decisions use resource usage and estimated state of charge; the reported evaluation found 23% fewer event rejections, 83% fewer deadline violations, and approximately 59% lower cluster energy than the compared schedulers [1]. Liu *et al.* studied heterogeneous multicore edge scheduling by combining task prioritization, core-aware mapping, and predictive DVFS. Their real-platform evaluation reported a 20.9% energy reduction with a 2.4% deadline-miss rate under the evaluated conditions [2].

Learning-based control is another active direction. DVFO jointly selects CPU, GPU, and memory frequencies and the fraction of DNN feature maps offloaded to the cloud. Across its evaluated devices,

models, and datasets, it reported an average 33% energy reduction and latency reductions ranging from 28.6% to 59.1% [3]. SparseDVFS uses operator sparsity to select frequency settings for CPU, GPU, and memory components; its authors reported a 78.17% average energy efficiency gain over the evaluated state-of-the-art solutions [4]. MetaDVFS instead transfers metadata-guided knowledge across device-application pairs and reports up to 17% improvement in the performance-power ratio and 70.8% faster adaptation in its evaluation [5]. These results illustrate the promise of learning, as well as the importance of workload and hardware context.

Offloading research treats computation placement as a joint communication and computation problem. An RMAB-based study develops scalable online policies for multi-access edge computing while accounting for mobility, heterogeneous resources, and handovers [6]. A 2026 review identifies persistent issues involving dynamic topologies, load balancing, security, privacy, and validation at large scale [7]. The AAEO framework combines learning and adaptive allocation for QoE-aware offloading; its reported evaluation claims up to 35% QoE improvement and 40% energy reduction, but these values are specific to the experimental setup [8].

Hardware studies show why specialized parallel data paths matter. A heterogeneous FPGA accelerator for image classification based on MobileNetV2 reported 7.35 W power consumption in its evaluated implementation [9]. EdgeLLM combines a CPU with an FPGA, mixed-precision processing, structured sparsity, and an end-to-end compiler; compared with the specified commercial GPU baseline, it reported $1.91\times$ throughput and $7.55\times$ energy efficiency [10]. These studies are promising; however, the benefits of accelerators depend strongly on the model structure, memory traffic, precision, compiler support, and comparison platform.

ENERGY-EFFICIENT TECHNIQUES

Energy-Aware Task Scheduling

Energy-aware scheduling extends conventional deadlines or makespan optimization by including power, battery state, thermal status, and energy budgets in the scheduling objective. A scheduler may place critical path tasks on high-performance cores and background work on efficiency cores, consolidate work to enable idle states, or use slack time to select a lower-power operating point. For edge devices, this is useful because it can reduce energy without requiring changes to the application model. Liu *et al.* demonstrated the value of combining task priority, processor awareness, and predictive DVFS, rather than treating scheduling and frequency selection as separate decisions [2]. Llorens-Carrodegua provided a complementary cluster-level example in which remaining battery estimates influence service placement [1]. Remaining limitations include dependence on accurate workload and energy models, scheduling overhead, and difficulty handling rapidly changing task arrivals.

Dynamic Voltage and Frequency Scaling

The DVFS changes the processor voltage and frequency according to the workload demand. It is effective when tasks have slack or when parallel resources are underutilized; however, aggressive reduction can increase execution time and deadline risk. DVFS also interacts with memory and accelerator frequencies; controlling only the CPU may render other components inefficient. The DVFO addresses this by co-optimizing several edge device frequencies with partial offloading [3]. SparseDVFS exploits the observation that dense compute-bound operators and sparse memory-bound operators may benefit from different frequency settings [4]. MetaDVFS explores the transfer of learned policies across applications and devices [5]. The key limitation is that DVFS policies must account for the transition overhead, leakage, thermal behavior, and energy cost of the controller itself.

Task Offloading and Resource Allocation

Offloading moves all or part of a task from a device to an edge server or the cloud. The decision depends on input size, required cycles, channel quality, transmission power, server load, deadline, and privacy requirements. Partial offloading is particularly relevant to parallel workloads because feature

extraction or preprocessing may remain local, whereas selected stages are transferred. RMAB formulations provide scalable policies for dynamic multi-access systems [6]. The AAEO shows how QoE and energy can be combined in an adaptive offloading objective [8]. However, communication energy, retransmissions, contention, mobility, and data protection requirements can eliminate the expected benefits. Therefore, offloading should be evaluated using end-to-end energy rather than only device CPU energy.

AI/ML-Based Optimization

AI and ML methods learn policies from workload, device, and network observations. DRL is attractive when the decision space includes discrete placement choices and continuous frequencies or transmit powers. The DVFO uses DRL for joint frequency control and feature-map offloading [3], whereas the MetaDVFS uses metadata to transfer knowledge across tasks [5]. Learning can reduce manual tuning and adapt to nonstationary workloads; however, training and inference consume energy and may require data that are unavailable during deployment. Policies can also generalize poorly across devices, operating systems, and workload distributions [11]. Therefore, a learning-based optimizer should be compared with lightweight heuristics and evaluated for its net energy benefit, including the controller overhead [12].

Hardware Accelerators

FPGAs, GPUs, NPU, and custom accelerators exploit parallelism and data reuse more directly than general-purpose CPUs. Techniques such as pipelining, quantization, mixed precision, weight reuse, and structured sparsity can reduce computation and memory traffic. The MobileNetV2 FPGA study demonstrated a practical low-power implementation [9], whereas EdgeLLM showed how CPU-FPGA cooperation can support large-model inference [10]. The limitations are substantial: accelerators may be underutilized on irregular workloads, require specialized compilation, and incur data-transfer energy. Their benefits must be measured using a clear workload, precision, memory, and baseline descriptions [11].

Mathematical Optimization

Mathematical optimization formulates task placement, processor selection, frequency, transmit power, and server allocation as constrained problems. Mixed-integer and nonlinear models can provide a useful reference optimum for small instances, whereas relaxations, decomposition, Lyapunov methods, and heuristics are more suitable for online control applications. A Sensors study derived a mathematical method to choose the fraction of jobs assigned to different edge servers while minimizing the average delay and energy [11]. The advantage is interpretability and explicit constraint handling. The drawback is complexity: exact solutions may be too slow for dynamic systems, and simplified models may omit memory contention, transition overhead, or measurement noise [12].

COMPARATIVE ANALYSIS

Reported improvements are study-specific and should not be interpreted as directly comparable because the evaluated hardware, workloads, baselines, and metrics differ across studies (Table 1).

Overall Findings

Three important observations emerge from the comparison. First, integrated approaches generally provide a more comprehensive optimization strategy because scheduling, DVFS, offloading, and resource allocation influence one another. Second, the evaluation environment has a major influence on the reported results; improvements obtained on a particular board, accelerator, workload, or DNN model cannot be automatically expected on another platform. Third, hardware-level and cross-platform validation remains an important research requirement, particularly for AI-based techniques, the training and inference overheads of which may reduce their practical energy benefits.

Overall, the comparative analysis indicates that future energy-efficient parallel computing systems should move from the isolated optimization of individual components toward **cross**-layer coordination of scheduling, DVFS, offloading, hardware acceleration, and resource allocation. Such systems should evaluate energy consumption, latency, reliability, privacy, thermal constraints, and controller overhead. This observation provides the basis for the research gaps and emerging research directions discussed in the following sections.

CHALLENGES

Hardware Heterogeneity is a Primary Challenge: CPUs, GPUs, FPGAs, and NPU expose different frequency controls, memory systems, and programming interfaces that must be addressed. Dynamic workloads further complicate control because an energy-optimal decision can change when task arrivals, data sizes or network conditions change. There is also a persistent performance-energy trade-off. Lower frequency may reduce dynamic power but extend execution time, whereas offloading may save local computation energy but add transmission cost and delay.

Learning-based methods introduce additional challenges. Training, policy inference, online adaptation, and data collection consume resources, and a policy trained on one device may not transfer reliably to another device. Security and privacy mechanisms also add computation and communication overhead, yet they are often treated separately from energy optimization. Finally, large deployments create coordination and scalability problems: a method that is practical for one device or a small cluster may become too costly when many heterogeneous nodes share limited network and computing resources.

EMERGING RESEARCH DIRECTIONS

Potential directions include the joint optimization of scheduling, DVFS, offloading, and resource allocation so that local and network energy are evaluated together. Hardware-software co-design is another promising direction; compilers, runtime systems, accelerators, and frequency controllers can expose shared information about sparsity, memory traffic, and deadlines. Transfer learning may reduce adaptation costs for DVFS and scheduling across related devices, whereas sparse-aware execution may reduce unnecessary operations in inference workloads.

Table 1. Summary of techniques and their limitations.

Technique	Representative work	Main objective	Reported benefit	Main limitation
Energy-aware scheduling	Liu <i>et al.</i> [2]	Heterogeneous mapping with predictive DVFS	20.9% lower energy; 2.4% deadline-miss rate	Depends on workload and platform models
Cluster scheduling	Llorens-Carrodegua [1]	Battery-aware placement in SBC clusters	23% fewer rejections; 83% fewer deadline violations; ~59% lower energy	Cluster-specific evaluation
DVFS and offloading	DVFO [3]	Joint CPU/GPU/memory control and partial offloading	33% average energy reduction; 28.6–59.1% latency reduction	DNN and device dependent
Sparse-aware DVFS	SparseDVFS [4]	Frequency control using operator sparsity	78.17% energy-efficiency gain in evaluation	Requires sparsity-aware profiling
Transfer-learning DVFS	Yan, and He [5]	Cross-device policy adaptation	Up to 17% performance-power improvement	Transfer quality varies by task
Offloading/resource allocation	Hou <i>et al.</i> [6]	Scalable dynamic MEC offloading	Online marginal-cost policies	Network and mobility assumptions
FPGA acceleration	Pérez <i>et al.</i> [9]	Low-power CNN inference	7.35 W in evaluated implementation	Specialized hardware and model
CPU-FPGA acceleration	EdgeLLM [10]	Efficient edge LLM inference	1.91× throughput; 7.55× energy efficiency	Comparison is platform-specific

Federated learning can support privacy-aware distributed optimization by sharing model updates rather than raw application data, although update communication and local training energy must be included in the analysis. Energy-aware accelerators, serverless or containerized edge platforms, renewable-energy-aware scheduling, and 6G-enabled edge intelligence are emerging directions. These are potential research avenues, not established outcomes; each requires hardware-level validation and a clear measurement methodology.

RESEARCH GAPS

First, real-world and hardware-level validation remains limited. Simulations can omit memory contention, frequency-transition delay, thermal effects, and measurement noise; the Llorens-Carrodegua study and the ODROID-N2+ hardware-in-the-loop work illustrate the value of physical evaluation [1, 12]. Second, ML and DRL methods often generalize poorly across devices and workloads. Yan and He explicitly addressed transfer, but a broader evaluation is still needed [5]. Third, the field lacks standardized benchmarks, workload traces, power instrumentation, and reporting conventions, which prevents reliable comparison of the reported percentages.

Fourth, scheduling, DVFS, offloading, and resource allocation are frequently optimized separately, even though their decisions interact. Fifth, security and privacy are commonly added as constraints after energy optimization rather than being co-optimized. Sixth, scalability to large heterogeneous deployments is insufficiently demonstrated. Seventh, the training and inference overhead of an ML controller can offset its savings. Finally, unified hardware-software co-design is still uncommon; accelerator architecture, compiler decisions, operating system scheduling, and network placement are often evaluated in isolation.

CONCLUSION

Energy-efficient parallel computing at the edge is becoming increasingly important because edge devices are required to process computationally intensive workloads under strict energy, latency, thermal, and resource constraints. This review examined six major approaches for improving energy efficiency: energy-aware task scheduling, dynamic voltage and frequency scaling (DVFS), task offloading and resource allocation, AI/ML-based optimization, hardware acceleration, and mathematical optimization. Comparative analysis shows that each approach can provide significant benefits when applied to an appropriate workload and a suitable computing environment. However, the effectiveness of these techniques depends strongly on the hardware platform, workload characteristics, network conditions, optimization objectives, and evaluation methodology.

The reviewed studies also demonstrate that optimizing a single system component is often insufficient for practical edge environments. Scheduling decisions influence processor utilization, DVFS affects both power and execution time, and offloading introduces communication energy and latency. Similarly, AI-based methods can improve adaptability but introduce computational and training overhead, whereas hardware accelerators can provide high efficiency for suitable workloads but may have limited flexibility and portability. These interactions indicate that energy efficiency should be considered a system-level objective rather than an isolated hardware or software problem.

Several research gaps remain. Real-world hardware validation is still limited, and differences in benchmarks, workloads, measurement techniques, and reporting methods make it difficult to directly compare results across studies. Learning-based approaches also require stronger cross-device and cross-workload generalization, and their controller overhead must be included when evaluating their actual energy benefits. In addition, security, privacy, thermal constraints, scalability, and reliability are not yet consistently optimized together with energy consumption. The lack of unified hardware-software approaches further limits the practical deployment of the existing techniques.

Therefore, future research should focus on integrated cross-layer optimization that jointly considers scheduling, DVFS, task offloading, resource allocation, and hardware acceleration. Hardware-software co-design, transfer learning, sparse-aware execution, federated optimization, energy-aware accelerators, and renewable-energy-aware edge systems represent promising directions. These approaches should be validated on physically heterogeneous platforms using standardized workloads and consistent end-to-end energy measurements.

In conclusion, no single technique can provide an optimal solution for all edge-computing scenarios. The future of energy-efficient parallel computing lies in adaptive and coordinated systems that can dynamically balance energy consumption, performance, latency, reliability, privacy, thermal constraints, and implementation overhead. The development and validation of such integrated approaches can enable more sustainable, responsive, and scalable edge-computing systems.

REFERENCES

1. Llorens-Carrodegua A, Sagkriotis SG, Cervelló-Pastor C, Pezaros DP. An energy-friendly scheduler for edge computing systems. *Sensors (Basel)*. 2021 Oct 28;21(21):7151.
2. Liu Y, Qu H, Chen S, Feng X. Energy efficient task scheduling for heterogeneous multicore processors in edge computing. *Sci Rep*. 2025 Apr 7;15(1):11819.
3. Zhang Z, Zhao Y, Li H, Lin C, Liu J. DVFO: learning-based DVFS for energy-efficient edge-cloud collaborative inference. *IEEE Trans Mob Comput*. 2024;23(10):9042–9059.
4. Zhang Z, Wu Z, Liu J, Mottola L. SparseDVFS: sparse-aware DVFS for energy-efficient edge inference [Preprint]. 2026. arXiv:2603.21908.
5. Yan J, He F, Sang Q, Tong B, Sun P, Gong Y, et al. Metadata-guided adaptable frequency scaling across heterogeneous applications and devices [Preprint]. 2025. arXiv:2509.22707.
6. Hou L, Li S, Shen Z, Fu J, Wu J, Jin J. Energy efficient offloading policies in multi-access edge computing systems with task handover. *IEEE Trans Mob Comput*. 2026 Jan 19:9419–9436. doi:10.1109/TMC.2026.3655433.
7. Mohapatra H. Task offloading and edge computing in IoT—gaps, challenges and future directions. *Comput Mater Contin*. 2026;87(3).
8. Nishad DK, Verma VR, Rajput P, Gupta S, Dwivedi A, Shah DR. Adaptive AI-enhanced computation offloading with machine learning for QoE optimization and energy-efficient mobile edge systems. *Sci Rep*. 2025 May 1;15(1):15263.
9. Pérez I, Figueroa M. A heterogeneous hardware accelerator for image classification in embedded systems. *Sensors (Basel)*. 2021 Apr 9;21(8):2637.
10. Huang M, Shen A, Li K, Peng H, Li B, Su Y, et al. EdgeLLM: a highly efficient CPU-FPGA heterogeneous edge accelerator for large language models. *IEEE Trans Circuits Syst I Regul Pap*. 2025;72(7):3352–3365.
11. Gelenbe E. Minimizing delay and power consumption at the edge. *Sensors (Basel)*. 2025 Jan 16;25(2):502.
12. Zhou T, Lin M. CPU frequency scheduling of real-time applications on embedded devices with temporal encoding-based deep reinforcement learning. *J Syst Archit*. 2023 Sep;142:102955.