ENGINEER

international scientific journal

ISSUE 2, 2025 Vol. 3
E-ISSN
3030-3893
ISSN
3060-5172

SLIB.UZ
Scientific library of Uzbekistan

A bridge between science and innovation

TOSHKENT DAVLAT TRANSPORT UNIVERSITETI
Tashkent state transport university

ENGINEER

A bridge between science and innovation

E-ISSN: 3030-3893

ISSN: 3060-5172

VOLUME 3, ISSUE 2

JUNE, 2025

engineer.tstu.uz

TASHKENT STATE TRANSPORT UNIVERSITY

ENGINEER

INTERNATIONAL SCIENTIFIC JOURNAL
VOLUME 3, ISSUE 2 JUNE, 2025

Founder of the international scientific journal “Engineer” – Tashkent State Transport University, 100167, Republic of Uzbekistan, Tashkent, Temiryo‘lchilar str., 1, office: 465, e-mail: publication@tstu.uz.

The “Engineer” publishes the most significant results of scientific and applied research carried out in universities of transport profile, as well as other higher educational institutions, research institutes, and centers of the Republic of Uzbekistan and foreign countries.

The journal is published 4 times a year and contains publications in the following main areas:

- Engineering;
- General Engineering;
- Aerospace Engineering;
- Automotive Engineering;
- Civil and Structural Engineering;
- Computational Mechanics;
- Control and Systems Engineering;
- Electrical and Electronic Engineering;
- Industrial and Manufacturing Engineering;
- Mechanical Engineering;
- Mechanics of Materials;
- Safety, Risk, Reliability and Quality;
- Media Technology;
- Building and Construction;
- Architecture.

Tashkent State Transport University had the opportunity to publish the international scientific journal “Engineer” based on the **Certificate No. 1183** of the Information and Mass Communications Agency under the Administration of the President of the Republic of Uzbekistan. **E-ISSN: 3030-3893, ISSN: 3060-5172.** Articles in the journal are published in English language.

The application of energy-saving technologies in parallel computing systems

M. Sultonov[1][a], B. Akmuradov[1][b]
[1]University of Management and Future Technologies, Tashkent, Uzbekistan

Abstract: This scientific paper provides a detailed analysis of the architecture of parallel computing systems, their operational principles, and methods for organizing parallel processing. In particular, it investigates the dynamic performance and energy consumption of the OpenBLAS DGEMM function using software tools and applications designed to evaluate energy usage on multi-core processors. Throughout the study, effective methods for ensuring energy efficiency are developed, and mathematical models for predicting energy consumption are applied to assess and optimize the energy costs associated with computational workloads. Additionally, the paper analyzes the performance metrics of high-efficiency parallel computing blocks and their impact on energy efficiency. Based on the research findings, practical recommendations are proposed to reduce energy consumption in parallel systems without compromising computational performance.

Keywords: processors, energy consumption, DVFS, CUDA, parallel computing, OpenBLAS DGEMM, energy-efficient technologies, energy efficiency

1. Introduction

Ensuring energy efficiency in parallel computing systems has become one of the most pressing scientific and practical challenges today. Systems based on supercomputers, cloud platforms, and multi-core processors are increasingly being utilized due to their high computational capacity. However, these systems also consume substantial amounts of electrical energy, which not only increases operational costs but also directly affects overall system efficiency.

High energy consumption can lead to technical failures and reduced hardware lifespan, in addition to exerting a negative impact on the environment. In the context of growing global demand for sustainable development and green technologies, the design and deployment of energy-efficient parallel computing systems are considered essential steps toward preserving ecological balance and reducing carbon footprints.

Moreover, effective energy management plays a critical role during system scalability. As additional nodes or cores are integrated, it is crucial to maintain a balance between performance and energy consumption; otherwise, system productivity may decline. Therefore, enhancing energy efficiency has become an integral part of technological advancement, scientific research, and industrial production.

In Uzbekistan, several government resolutions have been adopted to support the optimization of parallel computing systems and energy efficiency. For instance, Resolution No. PQ-3981 dated October 23, 2018, highlights the low level of implementation of resource- and energy-saving technologies and the slow pace of infrastructure modernization, which contribute to increased technological losses. Similarly, Presidential Decree No. PF-5527 dated August 28, 2018, established a national commission to address energy efficiency and the development of renewable energy sources. Additionally, Decree No. PF-6079 dated October 5, 2020, under the "Digital Uzbekistan – 2030" strategy, outlines initiatives for developing e-government and integrating state bodies into information systems. These policies and decrees set forth measures aimed at optimizing parallel computing systems and improving energy efficiency [1–3].

2. Research methodology

The Importance of Energy Efficiency

Modern computing infrastructures - including supercomputers, multi-core processors, cloud computing platforms, and artificial intelligence systems - are becoming increasingly complex and resource-intensive. Among the most critical resources required by these systems is electrical energy. Energy efficiency refers to the amount of energy consumed by a system to perform a specific computational task. This metric is of vital importance not only from a technical standpoint but also from economic, environmental, and strategic perspectives.

Firstly, energy efficiency reduces the operational costs of computing systems. Data centers, industrial computing complexes, and scientific analysis hubs operate using numerous servers and processors, which demand continuous electrical power and cooling systems. The more efficiently a system operates, the less energy it consumes, significantly lowering service costs. This constitutes a key economic advantage for large IT companies, scientific institutions, and cloud service providers.

Secondly, energy savings are essential for environmental protection. High energy consumption often increases the load on coal- or gas-based power plants, resulting in greater emissions of greenhouse gases such as CO_2. Therefore, energy-efficient computing systems are regarded as “green technologies.” With many countries and companies striving to adopt “carbon-neutral” or “zero-emissions” policies, the demand for energy-saving computing solutions is growing rapidly.

Thirdly, energy efficiency is a prerequisite for scalability and sustainable development. Modern parallel systems are expanding to include thousands of cores and nodes. Each additional node or core increases energy consumption, and without effective management, overall system performance

[a] https://orcid.org/0009-0003-4781-1601
[b] https://orcid.org/0009-0009-0661-6977

may decline. For this reason, advanced algorithms, hardware innovations, and software strategies are being developed to maintain an optimal balance between energy consumption and processing speed. This, in turn, lays the foundation for the long-term growth and sustainability of parallel computing.

Analysis of Research

In 2020, information and communication technologies accounted for 7% of global electricity consumption, and forecasts suggest that by 2030, this figure will reach a moderate level that aligns with projected scenarios ranging from 7% to 21%. This trend turns the need to improve the energy efficiency of digital platforms into a major new technological challenge.

There are two main approaches to addressing this challenge: hardware-based and software-based. The first approach focuses on developing energy-efficient hardware at the transistor level, aiming to design electronic devices that consume as little energy as possible.

The second approach targets the development of energy-efficient software. At the solution level, this can be further divided into system-level and application-level approaches. The system-level approach seeks to optimize the execution environment rather than the application itself. This currently represents the dominant method and includes techniques such as Dynamic Voltage and Frequency Scaling (DVFS), Dynamic Power Management (DPM), and energy-aware scheduling, all aimed at improving the energy efficiency of applications. DVFS manages the processor's clock frequency to reduce dynamic power consumption, which arises from switching activity in processor circuits. In contrast, static power is consumed when the processor is in an idle state. DPM reduces power usage by switching off electronic components or transitioning them to low-power states [4].

Application-level energy optimization methods involve adaptable solutions at the application layer and focus on optimizing the application itself rather than the system environment [2, 3]. One of the key solution variables in this category is workload distribution.

To ensure the reliability of the results presented in this work, a detailed statistical methodology was employed, involving multiple experimental runs to calculate the sample mean of the response variable. This methodology is based on the t-distribution and is used to compute the sample mean with a 95% confidence interval and a 0.05 (5%) margin of error. Moreover, the assumptions of normality and independence underlying the t-distribution are validated using Pearson's chi-squared test.

The book Metaheuristics: From Design to Implementation by Talbi serves as an excellent resource on the theoretical foundations and practical implementation of metaheuristic algorithms, covering the entire process from design to deployment.

The HCLWattsUp platform, developed by Fahad and Manumachu, offers an API for measuring energy consumption based on system-level physical power metrics. This provides a vital tool for accurately evaluating energy usage [5–7].

Libraries such as OpenBLAS and FFTW provide efficient linear algebra operations and fast Fourier transforms, enabling the optimization of core computational blocks in the field of high-performance computing (HPC).

Fahad and colleagues conducted a comparative study of energy measurement methods in computing systems, analyzing the strengths and limitations of each technique. The Top500 project, which ranks the world's most powerful supercomputers, offers valuable insights into global trends in the HPC domain. The OpenDwarfs benchmark suite, developed by Krommydas and collaborators, is described as an effective tool for evaluating applications across various architectures [8, 9].

The GHOST library provides highly efficient sparse linear algebra blocks, expanding capabilities for efficient computation on heterogeneous systems.

Domain decomposition methods proposed by Papadrakakis and colleagues introduce new opportunities for hybrid execution using both GPU and CPU architectures [10].

Khaleghzadeh and co-authors developed bi-objective optimization methods for running data-parallel applications on heterogeneous HPC platforms, focusing on both performance and energy efficiency. Their research is based on concrete algorithms and accounts for real-world performance profiles of platforms [11].

Lastovetsky and Reddy proposed data distribution methods based on functional performance models for heterogeneous systems, ensuring optimal allocation by considering the specific characteristics of networks and processors.

Lastovetsky and Twamley presented fundamental research aimed at developing more realistic models of network efficiency. Furthermore, novel data distribution algorithms by Khaleghzadeh and collaborators aim to optimize not only execution speed but also energy efficiency on HPC platforms, contributing significantly to the field [12–14].

The study by Rotem et al. provides a detailed description of the energy management system designed for Intel's Sandy Bridge microarchitecture. They highlight complex hardware and software approaches intended to enhance energy efficiency while maintaining peak performance. LIKWID, developed by Treibig, Hager, and Wellein, is a lightweight and efficient toolset designed for monitoring and analyzing performance in x86 multi-core environments. It allows for comprehensive assessment of resource utilization in multi-core systems [15].

AMD's uProf User Guide describes a tool that facilitates system-level efficiency and energy monitoring for AMD processors. In addition, AMD's BIOS and Kernel Developer's Guide (BKDG) presents essential technical details necessary for a deeper understanding of the Family 15h processor architecture and its interaction with the system. Hackenberg and co-authors compared power measurement techniques on standard compute nodes, evaluating the accuracy, reliability, and usability of various methods. This research helps identify the most appropriate approaches for practical power monitoring.

Intel's System Software Developer's Guide and Manycore Platform Software Stack (MPSS) for Xeon Phi coprocessors provide extensive information on software interfaces and tools necessary for efficient computing in multi-core and multiprocessor environments.

Scalability Challenges in Parallel Computing

The strength of parallel computing systems lies in their ability to distribute computational tasks across multiple cores or nodes, enabling faster and more efficient execution.

As a result, parallel computing approaches are extensively applied in fields such as large-scale scientific computations, artificial intelligence, meteorology, and genomics. However, increasing the number of cores or nodes does not always lead to proportional gains in computational performance. This issue gives rise to a complex set of scalability challenges that can hinder optimal system performance at technical, software, and energy levels.

Load Imbalance. In parallel computing, tasks are distributed among cores or nodes. However, if this distribution is not optimized, some nodes may be heavily overloaded while others remain underutilized or idle. For instance, one node may be operating at 80% capacity while another utilizes only 10%. This imbalance leads to a "bottleneck" effect - where the slowest-performing node dictates the overall execution time of the system. It also reduces energy efficiency, as even idle or underutilized resources consume power without contributing to useful computation.

Communication and Synchronization Overhead. As the system scales by adding more cores or nodes, the requirements for inter-node communication and synchronization increase. Each parallel process, upon completing its portion of the task, must exchange results with other nodes or integrate its output into a collective result. This leads to network congestion, communication delays, and synchronization stalls. Initially, when the number of cores is small, communication overhead is minimal and the system performs efficiently. However, as the number of nodes increases, communication begins to dominate computation time - a phenomenon known as the "communication bottleneck" - which severely degrades system efficiency.

Energy Scaling Inefficiency. One of the most critical aspects of scaling parallel systems is the energy-performance trade-off associated with each added core or node. Ideally, each new resource should contribute to increased performance. However, beyond a certain point (e.g., beyond 64 cores), additional nodes often yield diminishing performance returns while continuing to consume energy. Consequently, the system may consume more power without any noticeable improvement in performance - or may even perform worse. This transition from "superlinear scalability" to "sublinear" or "negative scalability" illustrates a point where further scaling becomes detrimental rather than beneficial.

Software and Architectural Limitations. Not all computational applications are infinitely scalable. Some algorithms are inherently unsuitable for parallel execution, or include sequential stages that must be executed in order. This is governed by the principle known as Amdahl's Law, which states that the maximum speedup of a system is limited by the sequential portion of the workload. Moreover, hardware architecture also plays a significant role in scalability. Elements such as cache memory, network topology, memory access patterns, and other shared resources may only perform efficiently up to a certain scale, after which performance gains diminish or reverse.

Challenges Associated with Heterogeneous Systems in Parallel Computing

Heterogeneous computing systems consist of multiple types of computing resources, such as CPUs, GPUs, FPGAs, and DSPs, working together in a unified system. These systems are widely used in various fields, particularly in scientific computing, artificial intelligence, real-time signal processing, and simulation systems, to enhance performance. When leveraged effectively, heterogeneous architectures can achieve high computational power and energy efficiency. However, working with these systems presents a range of complex technical, software, and management-related challenges.

Architectural Incompatibilities Between Resources. The devices within heterogeneous systems differ architecturally - for example, CPUs are designed for complex control, while GPUs are optimized for massive parallel processing. These differences result in significant incompatibilities in memory models, execution mechanisms, approaches to sequential and parallel processing, energy consumption, and computational speeds. Consequently, achieving equal performance across different devices for the same task becomes challenging, complicating the optimal distribution of workloads.

Complexity in Programming and Adaptation. Programming for heterogeneous systems is a complex, time-consuming process that requires specialized knowledge. Each device may require a different programming language or platform (e.g., C/C++ for CPUs, CUDA or OpenCL for GPUs). Developers must not only design computational algorithms but also manage data transfers between devices, synchronize memory, and handle resource management. This increases the complexity of the development process and raises the likelihood of errors.

Efficient Workload Distribution and Coordination. Given that the capabilities of resources in heterogeneous systems differ, determining which tasks should be assigned to which device is not straightforward. Certain tasks may execute more efficiently on a GPU, while others may be better suited for a CPU. Poor workload distribution can slow down overall performance or fail to fully utilize the available resources. This negatively affects not only performance but also energy efficiency.

Data Transfer and Synchronization Delays. In heterogeneous systems, data is typically stored in separate memory regions. For example, CPUs and GPUs each have their own dedicated memory, and data transfer between them occurs via networks or bridges. This transfer process introduces delays, additional energy consumption, and can slow down computation. Furthermore, synchronization mechanisms are required to ensure data consistency across devices, further consuming time and resources.

Optimization and Monitoring Complexity. To maximize the efficiency of heterogeneous systems, continuous monitoring, workload analysis, and dynamic adjustment of system parameters are necessary. However, performance metrics, energy profiles, and synchronization mechanisms for different devices are not uniform. This makes it difficult to automate the management and optimization of the system, increasing the complexity of maintaining optimal system performance.

Energy Management Strategies in Parallel Computing

Modern parallel computing systems, with the increasing number of cores, processors, and clusters, are simultaneously driving up the demand for electrical energy. This is especially evident in large scientific centers, supercomputers, or cloud computing infrastructures, which operate on millions of transistors, resulting in high energy consumption. Therefore, efficient energy management is crucial not only from an economic and environmental perspective but also for ensuring the overall performance efficiency of the system. Below are some of the key energy management strategies employed in parallel computing systems.

Dynamic Voltage and Frequency Scaling (DVFS). DVFS technology enables dynamic adjustments to the processor's operating frequency and corresponding voltage. When the processor is not required to run at full power, its frequency is reduced, thereby lowering energy consumption. This approach is particularly effective for energy saving during low-load conditions or when nodes are temporarily idle. However, care must be taken when using DVFS, as reducing frequency also results in a decrease in processing speed.

Energy-aware Task Scheduling. In this strategy, system tasks are analyzed based on their energy and performance requirements before execution. Tasks that consume less energy are assigned to more energy-efficient devices, while resource-intensive tasks are allocated to higher-performance nodes. For instance, simple tasks such as data sorting can be executed on energy-efficient CPUs, while computationally intensive tasks such as graphics processing are offloaded to GPUs. This approach not only conserves energy but also ensures optimal utilization of system resources.

Active and Passive Node Management (Power Gating and Sleep Modes). Parallel systems often consist of tens or even hundreds of nodes, but not all of them operate continuously. Unnecessary or idle nodes can be automatically turned off or put into sleep mode, leading to significant energy savings. This strategy is known as "power gating" or "sleep states." For example, in data centers that operate 24/7, some nodes may be temporarily shut down during off-peak hours, thus preventing excess heat generation and reducing cooling costs.

Optimizing Data Transfer and Memory Management. Communication and memory processes in parallel systems are as important as computation itself. As data exchange between systems increases, so does energy consumption, especially when large datasets are transmitted across long distances. Thus, energy can be saved by designing algorithms that maximize local memory usage, minimize data transfer, and reduce latency. Methods such as memory simplification and lazy writing are also effective strategies.

Automated Optimization and Intelligent Management Systems. Recently, AI-based energy management systems have been developed. Using machine learning (ML) or artificial intelligence (AI), these systems analyze real-time system loads and determine how energy can be saved. Based on historical data, these systems predict and optimally distribute energy. For example, the system can predict which nodes are likely to be more active and keep them in an active state while transitioning others to a waiting mode. While complex, this strategy represents one of the most advanced approaches to energy management.

3. Results

Results of Research and Analysis on Energy-Efficient Solutions

The acceleration and expansion of methods for optimizing energy consumption and performance are critical for ensuring energy efficiency and service quality in modern HPC platforms and cloud computing infrastructures. To achieve this expansion, it is essential to identify the key elements required, which involves reviewing the main stages of optimization methods.

The first step involves modeling hybrid applications, especially when different computing devices exist within the execution environment. A hybrid application consists of multiple multi-threaded cores, which execute simultaneously on different computing devices within the platform. The load of one core may significantly affect the performance of other cores due to intense competition for resources, a consequence of the close integration of the devices. Thus, modeling the performance and energy consumption of each core in hybrid applications is a complex issue.

At the same time, the research above considers configurations of hybrid applications that involve no more than one core per device. Each group of cores can be modeled as a single executing core, resulting in the platform being composed of various heterogeneous abstract processors. Grouping aims to minimize competition and interdependencies between abstract processors. Additionally, the overall resource utilization within the group is maximized, while between groups, it is minimized.

Thus, a hybrid application is presented as a set of computation cores executed within core groups, which are referred to as heterogeneous abstract processors. As an example, consider the platform depicted in Figure 1, which consists of two multi-core processors: a 24-core Intel Haswell processor with 64GB of memory, and a 22-core Intel Skylake processor. The first multi-core processor has two accelerators: an Nvidia K40c graphics processor and an Intel Xeon Phi 3120P. The second multi-core processor contains an Nvidia P100 PCIe graphics processor. Thus, the hybrid application executed on this platform is modeled by four heterogeneous abstract processors: CPU_1, GPU_1, PHI_1, and GPU_2. CPU_1 consists of 22 (out of 24) processor cores, GPU_1 is an Nvidia K40c graphics processor with a central processor core connected via a separate PCI-E channel, PHI_1 represents the Intel Xeon Phi processor with its central processor core connected through a separate PCI-E channel, and GPU_2 refers to the Nvidia P100 PCIe graphics processor, with its central processor core connected via a separate PCI-E channel.

Next, the performance and dynamic energy profiles of the computational cores are modeled using processor clock frequencies and system-level energy meters, based on a "ground-truth" methodology.

Finally, taking into account either the performance or dynamic energy profiles (or both), the data partitioning algorithm solves either a single-objective optimization problem for performance or energy, or a multi-objective optimization problem for both energy and performance, identifying the optimal Pareto solutions (load distributions) that minimize execution time and energy consumption during parallel execution.

However, two issues hinder the expansion of the proposed optimization methods. These issues are demonstrated by solving the two-objective optimization problem for energy and performance in heterogeneous processors with p errors.

First, building performance and dynamic energy profiles using system-level energy meters and a "ground-truth" methodology is sequential and costly. In the two applications, DGEMM and 2D-FFT, building discrete performance and dynamic energy profiles for workloads of sizes 210 and 256 takes 8 hours and 14 hours, respectively. The profiling procedure is carried out on the Intel Skylake processor. In brief, although the "ground-truth" method offers the highest accuracy, it is the most expensive method. Moreover, it cannot be used in environments without energy meters on the nodes.

Second, the data partitioning algorithm is executed sequentially, and for intermediate values of p, the execution time is still very large. For example, consider HEPOPTA, which solves the two-objective optimization problem for two scientific applications - matrix multiplication (DGEMM) and two-dimensional Fast Fourier Transform (2D-FFT) - executed on a hybrid platform. HEPOPTA runs sequentially on a single core of the Intel Skylake multi-core processor. For the DGEMM application, the execution time of the data partitioning algorithm varies from 4 seconds to 6 hours for p values between 12 and 192. For the 2D-FFT application, the execution time increases from 16 seconds to 16 hours for p values between 12 and 192.

Thus, there are three main challenges related to accelerating and expanding optimization methods on modern hybrid HPC platforms:

Accelerating sequential optimization algorithms, which ensures the rapid calculation of Pareto-optimal solutions from the perspective of optimizing performance and energy consumption.

For multi-core processors and accelerators, energy consumption measurement software sensors are used, with prediction models utilizing model variables that comply with high-additivity and energy conservation laws, as well as statistical tests such as high positive correlation.

Rapid construction of performance and dynamic energy profiles using software energy sensors.

All three challenges remain open problems. However, significant progress has been made in the development of software energy sensors for multi-core processors. For example, a software energy sensor for Intel multi-core processors can be implemented using a linear model for energy prediction based on resource utilization variables and performance monitoring counters (PMCs), which has shown an accuracy of 10-20% for popular scientific cores.

Previous generations of Nvidia graphics processors were poorly equipped for modeling energy consumption during runtime. However, the latest generation of graphics processors, such as the Nvidia A40, improves the support for energy meters, making it easier to model energy consumption during runtime.

In this work, we examined application-level optimization methods that address issues specific to modern HPC platforms. The application of these methods requires an energy profile for the computation cores (components) of hybrid parallel applications executed on different computing devices within HPC platforms. Therefore, we summarized the three main methods for energy measurement at the component level and provided the trade-offs in terms of accuracy and performance[21].

Finally, the expansion of energy and performance optimization methods is critical for ensuring energy efficiency and meeting service requirements in modern HPC platforms and cloud computing infrastructures. We presented the necessary building blocks for achieving this expansion and examined the challenges involved in this expansion. In brief, two important issues are rapid optimization methods and, particularly, accurately measuring the energy consumption over time for components operating in accelerators.

Currently, energy consumption modeling for components operating in accelerators is still in its early stages. To date, advances in energy consumption modeling have mainly focused on CPUs and not accelerators. Older generations of Nvidia graphics processors were poorly equipped for modeling energy consumption during runtime. However, the latest generation of graphics processors, such as the Nvidia A40, ensures good support for energy consumption modeling, making it easier to model energy consumption accurately.

We illustrate the results of this research with a highly optimized matrix multiplication program running on a hybrid computing platform. The program computes matrix multiplication based on the following formula:

$$C = \alpha \times A \times B + \beta \times C \qquad (1)$$

Here, A, B, and C are matrices of sizes $M \times N$, $N \times N$, and $M \times N$, respectively, and α and β are real numbers. The program calls functions from the CUBLAS library for Nvidia graphics processors and functions from the Intel MKL DGEMM library for Intel Xeon Phi and central processors. The versions of Intel MKL and CUDA used are 2017.0.2 and 9.2.148, respectively.

The platform consists of five hybrid processors: Intel Haswell E5-2670V3 multi-core processor, Intel Xeon Gold 6152 multi-core processor, Nvidia K40c graphics processor, Nvidia P100 PCIe graphics processor, and Intel Xeon Phi 3120P. Additionally, the platform includes five hybrid abstract processors, each executing a computation core: CPU_1, GPU_1, xeonphi_1, CPU_2, and GPU_2.

The following figure shows the execution time and dynamic energy functions of the processors as a function of workload size, ranging from 64×10^{112} to $19{,}904 \times 10^{112}$, with the first measurement scale for M set to 64. When the program is executed, static and dynamic energy consumption is measured using the Ground-Truth method. The execution time functions exhibit a continuous and strictly increasing pattern, while the energy functions clearly follow linear growth functions.

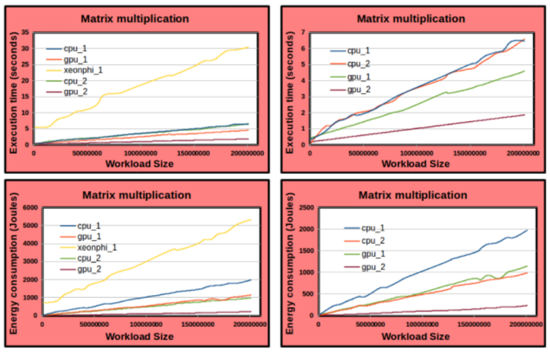

Figure 1. Execution Time and Energy Consumption

In this figure, the left column presents the execution time and energy consumption profiles for five hybrid processors used in the matrix multiplication program. The right column excludes the Xeon Phi profile, as its energy profile dominates the other energy profiles. It is important to note that the execution time profiles for CPU_1 and CPU_2 are very similar; however, the energy profile of CPU_1 is significantly higher than that of CPU_2.

The following figure illustrates the Pareto fronts obtained for two workload sizes, $12{,}352 \times 10^{112}$ and $15{,}552 \times 10^{112}$, for the matrix multiplication program using the proposed algorithms. These algorithms treat energy and performance profiles as linear functions. Each Pareto front consists of four linear segments, with the solution balancing the workload for minimal execution time[18].

Intel's multi-matrix processors (RAPL) allow for energy consumption management and frequency control by setting an average voltage limit. Additionally, previous processor generations such as Sandybridge and Ivybridge E5 use resource utilization and performance monitoring counters (PMC) to predict energy consumption.

However, for Haswell and later processors, RAPL utilizes separate voltage regulators to manage CPU and DRAM operations individually. VR IMON is an analog circuit in the voltage regulator (VR) that tracks current. However, there are delays between the measured signal and the actual current signal, which can affect accuracy. The CPU periodically measures these readings to compute energy consumption.

RAPL energy counters are provided in model-specific registers (MSR) with a device-specific list. These energy areas are categorized for precise platform control. These areas include the Package, consisting of both the primary and non-primary components; DRAM, which is available only for servers; and CPU cores and graphics processors.

The parallel program computes the N×N multiplication of two dense square matrices A and B, executed on two multi-matrix processors, Intel CPUs: Intel Haswell E5-2670V3 (CPU1) and Intel Xeon Gold 6152 (CPU2). The B matrix is replicated on both processors. CPU1 multiplies matrix A1 and B, while CPU2 multiplies A2 and B. The local matrix multiplications are computed using Intel MKL DGEMM.

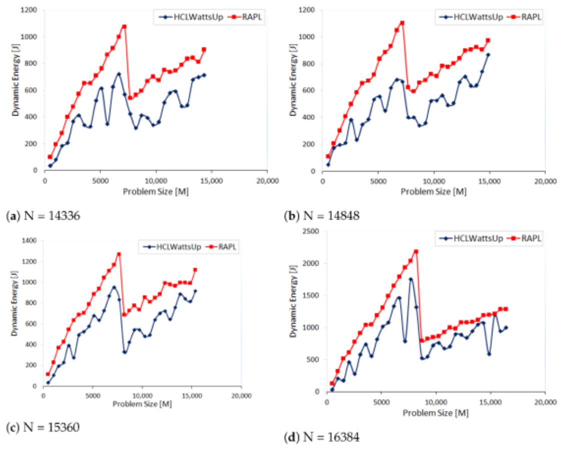

Figure 2. Dynamic Energy Consumption of DGEMM on Two Multi-Matrix Intel Processors

Matrix A is divided into two smaller sub-matrices, A1 and A2, of dimensions M×N and K×N, respectively. These sub-matrices are distributed across processors using a data partitioning algorithm based on the model. The algorithm takes as input the matrix size N and the dynamic energy functions of the processors, e1(x, y) and e2(x, y), where e_i(x, y) represents the energy consumption for multiplying matrices of size x×y and y×y. Thus, the dynamic energy function is expressed as a surface.

4. Discussion

The algorithm intersects the surfaces of the dynamic energy functions in a plane where N equals the matrix size. The intersection forms two curves. Subsequently, the algorithm selects two points, (M, e1(M, N)) and (K, e2(K, N)), where the sum of their energy consumption, e1(M, N) + e2(K, N), is minimized.

Intel RAPL represents a more dynamic energy consumption model for all workload sizes compared to actual data. The prediction errors of Intel RAPL are summarized in the following table[20].

Table 1

Comparison of Dynamic Energy Consumption for DGEMM: Intel RAPL Prediction Errors vs. Actual Data.

Workload Size (N)	Min	Max	Average
14,336	17%	172%	65%
14,848	12%	153%	58%
15,360	13%	240%	56%
16,384	2%	300%	56%

The data partitioning algorithm determines the distribution of the workload by using the workload size N and the dynamic energy profiles of the two processors as input data. Then, by distributing the workload, the dynamic energy consumption is obtained when the parallel program is executed.

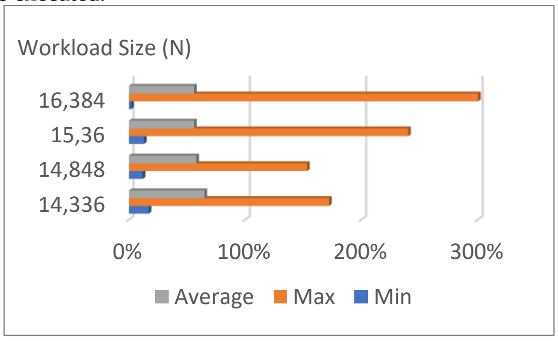

Figure 3. Comparison of Dynamic Energy Consumption Results

To organize energy-efficient parallel computing, specific architectural and software solutions must be developed. In this context, the coordinated operation of system components according to their energy consumption, the rational distribution of computational tasks, and dynamic workload management play a key role. In parallel computing environments, particularly on multi-core and GPU-based platforms, improving overall system performance can be achieved by optimizing task placement, reducing data exchange processes, and utilizing energy-efficient algorithms.

5. Conclusion

This research systematically examines existing approaches to improving energy efficiency in parallel computing and the key challenges associated with their large-scale application. The author emphasizes that the increasing complexity of computing systems - particularly the widespread use of heterogeneous architectures - has intensified the need for accurate and reliable energy consumption assessment and effective optimization. Specifically, finding a balance between performance and energy consumption in large-scale systems - i.e., saving energy while maintaining computational speed - has been highlighted as a critical issue.

While analyzing the results achieved using existing technologies and methods - such as Dynamic Voltage and Frequency Scaling (DVFS), workload balancing, and energy-aware programming models - attention has also been drawn to their limitations. These include the incomplete scalability of experimental methods used on multi-core and heterogeneous platforms for enhancing energy efficiency, the complexity of measurements, and the challenges in developing optimization models for systems operating in real environments.

Overall, this research provides a comprehensive analysis of fundamental and practical approaches to enhancing the energy efficiency of parallel computing systems, creating a solid scientific foundation for future research directions in this field. The main idea proposed is that to achieve real success, not only hardware-level solutions but also algorithmic, software, and system-level approaches must be integrated into comprehensive solutions. Therefore, this work serves as an essential and scientifically grounded source for specialists and system architects researching parallel and energy-aware computing.

Based on the findings of this study, the following proposals and recommendations have been developed:

1. Balancing Dynamic Performance and Energy Consumption: As demonstrated in the research using the OpenBLAS DGEMM program, optimizing the balance between dynamic frequency scaling and energy consumption can enhance processor performance and reduce energy consumption. Algorithms that automatically adjust the operating frequency based on workload conditions are recommended.

2. Optimizing Workload Distribution in Heterogeneous Systems: In multi-core and heterogeneous platforms (e.g., CPU-GPU combinations), it is crucial to optimally distribute workloads based on energy and performance metrics. By dividing data and tasks according to the functional performance model of cores, overall energy consumption can be reduced.

3. Utilizing Energy Prediction and Monitoring Tools: It is recommended to regularly monitor system-level energy measurements and dynamically manage performance modes based on the results. For example, energy APIs such as HCLWattsUp can be used to determine real-time energy consumption and select the most effective performance strategies.

4. Optimizing Parallel Programs: To enhance resource utilization in parallel programs, it is recommended to implement energy-efficient parallelization of algorithms, reduce unnecessary data exchanges, and maximize the proper loading of computational resources.

5. Adopting Energy-Efficient Algorithms and Technologies: The active implementation of energy-saving technologies in parallel computing systems, such as "power gating" and "dynamic voltage and frequency scaling" (DVFS), can significantly reduce energy consumption.

References

[1] Decree No. PQ-3981 of the President of the Republic of Uzbekistan, dated October 23, 2018, "On the Implementation of Resource and Energy-Efficient Technologies, Infrastructure Renewal, and the Reduction of Technological Losses."

[2] Decree No. PF-5527 of the President of the Republic of Uzbekistan, dated August 28, 2018, "On the Establishment of the National Commission for the Efficient Use of Energy and the Development of Renewable Energy Sources."

[3] Decree No. PF-6079 of the President of the Republic of Uzbekistan, dated October 5, 2020, "On the Development of E-Government and the Integration of State Agencies into Information Systems within the Framework of the 'Digital Uzbekistan-2030' Strategy."

[4] Talbi, E.G. Metaheuristics: From Design to Implementation; John Wiley & Sons: Hoboken, NJ, USA, 2009; Volume 74.

[5] Fahad, M.; Manumachu, R.R. HCLWattsUp: Energy API Using System-Level Physical Power Measurements Provided by Power Meters; Heterogeneous Computing Laboratory, University College Dublin: Dublin, Ireland, 2023.

[6] OpenBLAS. OpenBLAS: An Optimized BLAS Library. Available online: https://github.com/xianyi/OpenBLAS (accessed on 1 December 2022).

[7] Fahad, M.; Shahid, A.; Reddy, R.; Lastovetsky, A. A Comparative Study of Methods for Measurement of Energy of Computing. Energies 2019, 12, 2204.

[8] Top500. The Top500 Supercomputers List. Available online: https://www.top500.org (accessed on 1 December 2022).

[9] Krommydas, K.; Feng, W.C.; Antonopoulos, C.D.; Bellas, N. OpenDwarfs: Characterization of Dwarf-Based Benchmarks on Fixed and Reconfigurable Architectures. Journal of Signal Processing Systems 2016, 85, 373–392.

[10] Kreutzer, M.; Thies, J.; Röhrig-Zöllner, M.; Pieper, A.; Shahzad, F.; Galgon, M.; Basermann, A.; Fehske, H.; Hager, G.; Wellein, G. GHOST: Building Blocks for High Performance Sparse Linear Algebra on Heterogeneous Systems. International Journal of Parallel Programming 2016, 45, 1046–1072.

[11] Papadrakakis, M.; Stavroulakis, G.; Karatarakis, A. A New Era in Scientific Computing: Domain Decomposition Methods in Hybrid CPU–GPU Architectures. Computer Methods in Applied Mechanics and Engineering 2011, 200, 1490–1508.

[12] Khaleghzadeh, H.; Fahad, M.; Shahid, A.; Reddy, R.; Lastovetsky, A. Bi-Objective Optimization of Data-Parallel Applications on Heterogeneous HPC Platforms for Performance and Energy Through Workload Distribution. IEEE Transactions on Parallel and Distributed Systems 2021, 32, 543–560.

[13] Khaleghzadeh, H.; Reddy, R.; Lastovetsky, A. Efficient Exact Algorithms for Continuous Bi-Objective Performance-Energy Optimization of Applications with Linear Energy and Monotonically Increasing Performance Profiles on Heterogeneous High Performance Computing Platforms. Concurrency and Computation: Practice and Experience 2022, e7285.

[14] FFTW. FFTW: A Fast, Free C FFT Library. Available online: https://www.fftw.org (accessed on 1 December 2022).

[15] Lastovetsky, A.L.; Reddy, R. Data Partitioning with a Realistic Performance Model of Networks of Heterogeneous Computers. In Proceedings of the Parallel and Distributed Processing Symposium, Hong Kong, China, 13–15 December 2004.

[16] Lastovetsky, A.; Reddy, R. Data Partitioning with a Functional Performance Model of Heterogeneous Processors. International Journal of High Performance Computing Applications 2007, 21, 76–90.

[17] Lastovetsky, A.; Twamley, J. Towards a Realistic Performance Model for Networks of Heterogeneous Computers. In High Performance Computational Science and Engineering; Springer: Berlin, Germany, 2005; pp. 39–57.

[18] Khaleghzadeh, H.; Reddy, R.; Lastovetsky, A. A Novel Data-Partitioning Algorithm for Performance Optimization of Data-Parallel Applications on Heterogeneous HPC Platforms. IEEE Transactions on Parallel and Distributed Systems 2018, 29, 2176–2190.

[19] Khaleghzadeh, H.; Fahad, M.; Reddy, R.; Lastovetsky, A. A Novel Data Partitioning Algorithm for Dynamic Energy Optimization on Heterogeneous High-Performance Computing Platforms. Concurrency and Computation: Practice and Experience 2020, 32, e5928.

[20] Fahad, M.; Shahid, A.; Manumachu, R.R.; Lastovetsky, A. Accurate Energy Modelling of Hybrid Parallel Applications on Modern Heterogeneous Computing Platforms Using System-Level Measurements. IEEE Access 2020, 8, 93793–93829.

[21] Alexey Lastovetsky, Ravi Reddy Manumachu Energy-Efficient Parallel Computing: Challenges to Scaling, Information 2023, 14(4), 248.

Information about the author

Mukhamma-dali Sultonov	University of Management and Future Technologies universiteti magistranti. E-mail: sultonovmuhammadali226960@gmail.com Tel.: +998951232303 https://orcid.org/0009-0004-6341-1680
Bakhtiyor Akmuradov	University of Management and Future Technologies universiteti dotsenti, t.f.f.d., (PhD), E-mail: b.u.akmuradov@gmail.com Tel.: +99897890 47 57 https://orcid.org/0009-0008-0600-3604

CONTEXT / MUNDARIJA