Optimization of Data-Parallel Applications on Heterogeneous HPC Platforms for Dynamic Energy Through Workload Distribution

Hamidreza Khaleghzadeh(✉), Muhammad Fahad, Ravi Reddy Manumachu, and Alexey Lastovetsky

School of Computer Science, University College Dublin, Belfield, Dublin 4, Ireland
{hamidreza.khaleghzadeh,ravi.manumachu,alexey.lastovetsky}@ucd.ie, muhammad.fahad@ucdconnect.ie

Abstract. Energy is one of the most important objectives for optimization on modern heterogeneous high performance computing (HPC) platforms. The tight integration of multicore CPUs with accelerators in these platforms present several challenges to optimization of multithreaded data-parallel applications for dynamic energy.

In this work, we formulate the optimization problem of data-parallel applications on heterogeneous HPC platforms for dynamic energy through *workload distribution*. We propose a solution method to solve the problem. It consists of a data-partitioning algorithm that employs load imbalancing technique to determine the workload distribution minimizing the dynamic energy consumption of the parallel execution of an application. The inputs to the algorithm are discrete dynamic energy profiles of individual computing devices.

We experimentally analyse the proposed algorithm using two multithreaded data-parallel applications, matrix multiplication and 2D fast Fourier transform. The load-imbalanced solutions provided by the algorithm achieve significant dynamic energy reductions (on the average 130% and 44%) compared to the load-balanced ones for the applications.

Keywords: High performance computing · Heterogeneous platforms · Energy of computation · Multicore CPU · GPU · Xeon Phi

1 Introduction

Energy consumption is one of the main challenges hindering high performance computing (HPC) community from breaking the exascale barrier [9].

Supported by Science Foundation Ireland (SFI) under Grant Number 14/IA/2474.

Electronic supplementary material The online version of this chapter (https://doi.org/10.1007/978-3-030-48340-1_25) contains supplementary material, which is available to authorized users.

Energy optimization in HPC context is studied briefly in connection with bi-objective optimization for performance and energy. State-of-the-art solution methods for bi-objective optimization problem can be broadly classified into *system-level* and *application-level* categories. System-level solution methods aim to optimize performance and energy of the environment where the applications are executed. The methods employ application-agnostic models and hardware parameters as decision variables. The dominant decision variable in this category is Dynamic Voltage and Frequency Scaling (DVFS). Majority of the works in this category optimize for performance with energy budget as a constraint. Application-level solution methods proposed in [2,12–14] use application-level parameters as decision variables and application-level models for predicting the performance and energy consumption of applications. The application-level parameters include the number of threads, number of processors, loop tile size, workload distribution, etc. Chakraborti et al. [2] consider the effect of heterogeneous workload distribution on bi-objective optimization of data analytics applications by simulating heterogeneity on homogeneous clusters. The performance is represented by a linear function of problem size and the total energy is predicted using historical data tables. Research works [13,14] demonstrate by executing real-life data-parallel applications on modern multicore CPUs that the functional relationships between performance and workload distribution and between energy and workload distribution have complex (non-linear) properties. They target homogeneous HPC platforms.

Modern heterogeneous HPC platforms feature tight integration of multicore CPUs with accelerators such as graphical processing units (GPUs) and Xeon Phi coprocessors to provide cutting-edge computational power and increased energy efficiency. This has resulted in inherent complexities such as severe resource contention for shared on-chip resources (Last Level Cache, Interconnect) and Non-Uniform Memory Access (NUMA). One visible manifestation of these complexities is a complex functional relationship between energy consumption and workload size of applications executing on these platforms where the shape of energy profiles may be highly non-linear and non-convex with drastic variations. This, however, provides an opportunity for application-level energy optimization through workload distribution as a decision variable.

Consider the dynamic energy profiles of multithreaded matrix-matrix multiplication (DGEMM) and 2D fast Fourier transform (2D-FFT) application executed on two connected heterogeneous multi-accelerator NUMA nodes, HCLServer1 (Table 1) and HCLServer2 (Table 2). The multicore CPU in HCLServer1 is integrated with one Nvidia K40c GPU and one Intel Xeon Phi 3120P. The multicore CPU in HCLServer2 is integrated with one Nvidia P100 GPU. DGEMM computes the matrix product, $C = \alpha \times A \times B + \beta \times C$, where A, B, and C are respectively dense matrices of size $m \times n$, $n \times n$, and $m \times n$ and α and β are constant floating-point numbers. 2D-FFT computes the Fourier transform of a complex matrix of size $m \times n$.

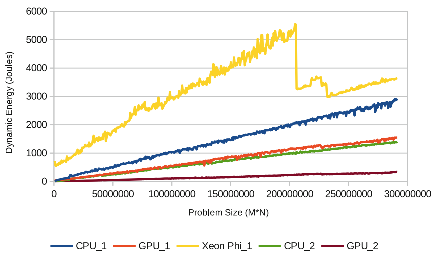
A data-parallel application executing on this heterogeneous platform, consists of a number of kernels (generally speaking, multithreaded), running in parallel on

Table 1. HCLServer1 specifications.

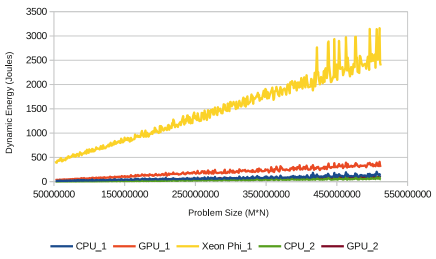
Intel Haswell E5-2670V3		Nvidia K40c		Intel Xeon Phi 3120P	
Socket(s), Cores per socket	2, 12	No. of processor cores	2880	No. of processor cores	57
Main memory	64 GB	Total board memory	12 GB	Total main memory	6 GB
Idle Power (W)	60	Idle Power (W)	68	Idle Power (W)	91

Table 2. HCLServer2 specifications.

Intel Xeon Gold 6152		Nvidia P100 PCIe	
Socket(s), Cores per socket	1, 22	No. of processor cores	3584
Main memory	96 GB	Total board memory	12 GB
Idle Power (W)	60	Idle Power (W)	30



(a) DGEMM



(b) 2D-FFT

Fig. 1. Dynamic energy functions for the five abstract processors on HCLServer1 and HCLServer2. (a) DGEMM, and (b) 2D-FFT.

different computing devices of the platform. In order to apply our optimization algorithms, each group of cores executing an individual kernel of the application is modelled as an abstract processor [21] so that the executing platform is represented as a set of abstract processors. HCLServer1 is modelled by three abstract processors, CPU_1, GPU_1, and PHI_1. CPU_1 represents 22 (out of total 24) CPU cores. GPU_1 involves the Nvidia K40c GPU and a host CPU core connected to this GPU via a dedicated PCI-E link. PHI_1 is made up of one Xeon Phi 3120P and its host CPU core connected via a dedicated PCI-E link. In the same manner, HCLServer2 is modelled by two abstract processors, CPU_2 and GPU_2. Since there should be a one-to-one mapping between the abstract processors and computational kernels, any hybrid application executing on the servers should consist of five kernels, one kernel per computational device.

The dynamic energy profiles for the applications are shown in the Fig. 1. Each profile presents the dynamic energy consumption of a given processor versus workload size executed on the processor. In the figure for 2D-FFT, the dynamic energy profile for Phi_1 is ignored since it consumes 10 times more energy and dominates the other profiles. The dynamic energy consumptions are measured using Watts Up Pro power meter. We will elaborate the practical methodology to construct the discrete dynamic energy profiles in a following section.

Consider the execution of DGEMM for the workload size 2496×10112 employing all the five abstract processors, {CPU_1,CPU_2,GPU_1,GPU_2, PHI_1}. The solution determined by load-balanced algorithm is {64,320,64, 2048,0} and its dynamic energy consumption is 84 J. The optimal workload distribution assigns the whole workload to GPU_2 resulting in dynamic energy consumption of 24 J and thereby providing 150% reduction in energy. Consider the execution of 2D-FFT for the workload size 9120×51200 (2D signal) employing all the five abstract processors. The solution (workload distribution) determined by load-balanced algorithm is {1200,5376,1024,1472,0} and its dynamic energy consumption is 82 J. The load-balancing algorithm employs horizontal decomposition of the rows of the 2D signal. The optimal workload distribution assigns the whole workload to CPU_2 resulting in dynamic energy consumption of 40 J and thereby providing 105% reduction in energy. Our proposed solution finds these optimal workload distributions.

In this work, we propose a novel data-partitioning algorithm, *HEOPTA*, that determines optimal workload distribution minimizing the dynamic energy consumption of data-parallel applications executing on heterogeneous platforms for the most general shapes of dynamic energy profiles of the participating processors. To model the performance of a parallel application and build its speed functions, the execution time of any computational kernel can be measured accurately using high precision processor clocks. There is however no such effective equivalent for measuring the energy consumption. Physical measurements using power meters are accurate but they do not provide a fine-grained decomposition of the energy consumption during the application run in a hybrid platform. We propose a practical methodology to determine this decomposition, which employs only system-level energy measurements using power meters. The methodology allows us to build discrete dynamic energy functions of abstract processors with sufficient accuracy for the application of HEOPTA.

We experimentally analyse the accuracy of our energy modelling methodology and the performance of HEOPTA using two data-parallel applications, DGEMM and 2D-FFT, on a cluster of two heterogeneous nodes. We show that the load-imbalanced solutions provided by the algorithm achieve significant dynamic energy reductions compared to the load balanced solutions.

Our main contribution of this work is a novel data-partitioning algorithm that determines optimal workload distribution minimizing the dynamic energy consumption of data-parallel applications executing on heterogeneous platforms for the most general shapes of dynamic energy profiles of the processors.

The paper is organized as follows. Section 2 presents related work. Section 3 presents the formulation of the heterogeneous dynamic energy optimization problem. Section 4 describes our algorithm solving the problem. In Sect. 5, the device-level approach for dynamic energy modelling is illustrated. Section 6 presents the experimental results. Finally, Sect. 7 concludes the paper.

2 Related Work

In this section, we will cover research works on bi-objective optimization for performance and energy and notable works model the energy of computation.

Analytical studies of bi-objective optimization for performance and energy are presented in [3,5,15]. Choi et al. [3] extend the energy roofline model by adding an extra parameter, power cap, to their execution time model. Drozdowski et al. [5] use iso-energy map, which are points of equal energy consumption in a multi-dimensional space of system and application parameters, to study performance-energy trade-offs. Marszalkowski et al. [15] analyze the impact of memory hierarchies on time-energy trade-off in parallel computations, which are represented as divisible loads. The works reviewed do not consider workload distribution as a decision variable.

Basmadjian et al. [1] constructs a power model of a server using the summation of power models of its components: the processor (CPU), memory (RAM), fans, and disk (HDD). A model representing the energy consumption of a multicore CPU by a non-linear function of workload size is developed in [13]. Nagasaka et al. [16] propose PMC-based statistical power consumption modelling technique for GPUs that run CUDA applications. Song et al. [20] present power and energy prediction models based on machine learning algorithms such as backpropagation in artificial neural networks (ANNs). Shao et al. [19] develop an instruction-level energy consumption model for a Xeon Phi processor.

3 Formulation of Heterogeneous Dynamic Energy Optimization Problem

Consider a workload size n executing on p processors with dynamic energy functions, $E = \{e_0(x), ..., e_{p-1}(x)\}$ where $e_i(x)$, $i \in \{0, 1, \cdots, p-1\}$, is a discrete dynamic energy function of processor P_i with a cardinality of m. The heterogeneous dynamic energy optimization problem can be formulated as follows:

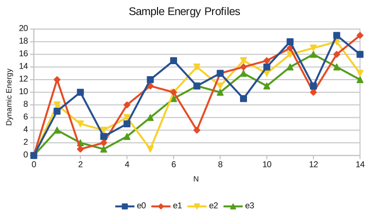
Heterogeneous Dynamic Energy Optimization Problem, *HEOPT*$(n$, p, m, E, X_{opt}, $e_{opt})$: The problem is to find a workload distribution, $X_{opt} = \{x_0, ..., x_{p-1}\}$, for the workload n executing on p heterogeneous processors so that the solution minimizes dynamic energy consumption during the parallel execution of n. The parameters (n, p, m, E) are the inputs to the problem. The outputs are X_{opt}, which is the optimal solution (workload distribution), and e_{opt}, which represents the dynamic energy consumption of the optimal solution. The formulation below is a integer non-linear programming (INLP) problem.

$$e_{opt} = \min_{X} \sum_{i=0}^{p-1} e_i(x_i) \qquad \text{Subject to} \quad \sum_{i=0}^{p-1} x_i = n, \tag{1}$$
$$\text{where} \quad p, m, n \in \mathbb{Z}_{>0} \quad \text{and} \quad x_i \in \mathbb{Z}_{\geq 0} \quad \text{and} \quad e_i(x) \in \mathbb{R}_{>0}$$

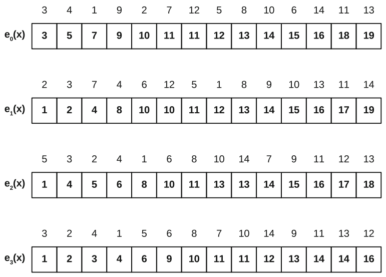
The objective function in Eq. 1 is a function of workload distribution X, $X = \{x_0, ..., x_{p-1}\}$, for a given workload n executing on the p processors. The number of active processors (processors that are assigned non-zero workload size) in the optimal solution (X_{opt}) may be less than p.

4 HEOPTA: Algorithm Solving HEOPT Problem

In this section, we will introduce HEOPTA, a branch-and-bound algorithm solving HEOPT. The bounding criteria in HEOPTA are *energy threshold* and *size threshold*, which are explained below.



(a) Sample dynamic energy functions



	3	4	1	9	2	7	12	5	8	10	6	14	11	13
$e_0(x)$	3	5	7	9	10	11	11	12	13	14	15	16	18	19
	2	3	7	4	6	12	5	1	8	9	10	13	11	14
$e_1(x)$	1	2	4	8	10	10	11	12	13	14	15	16	17	19
	5	3	2	4	1	6	8	10	14	7	9	11	12	13
$e_2(x)$	1	4	5	6	8	10	11	13	13	14	15	16	17	18
	3	2	4	1	5	6	8	7	10	14	9	11	13	12
$e_3(x)$	1	2	3	4	6	9	10	11	11	12	13	14	14	16

(b) Functions stored in arrays

Fig. 2. (a) Dynamic energy functions of a sample application executing on four heterogeneous processors. (b) The same functions stored in arrays.

First, the algorithm is informally explained using a simple example. Consider a workload $n = 12$ executing on a given platform consisting of four heterogeneous processors ($p = 4$). Figure 2 (a) shows the discrete dynamic energy functions, $E = \{e_0(x), \cdots, e_3(x)\}$, with a cardinality of 14 ($m = 14$), as inputs to HEOPTA. Figure 2 (b) shows the discrete dynamic energy functions which are stored as arrays in non-decreasing order of energy consumption.

To solve the HEOPT problem and find the optimal workload distribution, a straightforward approach is to explore a full solution tree in order to build all combinations and finally select a workload distribution that its dynamic energy consumption is minimum. The tree explored by such a naive approach is shown in Fig. 3 which contains all the combinations for $n = 12$ and $p = 4$. Due to the lack of space, the tree is shown partially.

The naive algorithm starts tree exploration from the root at the level L_0 of the tree. The root node is labelled by 12 which represents the whole workload to be distributed between 4 processors $\{P_0, P_1, P_2, P_3\}$. Then, fifteen ($= m + 1$) problem sizes, including a zero problem size along with all problem sizes in the dynamic energy function $e_0(x)$, are assigned to the processor P_0 one at a time. Therefore, the root is expanded into 15 children. The value, which labels an

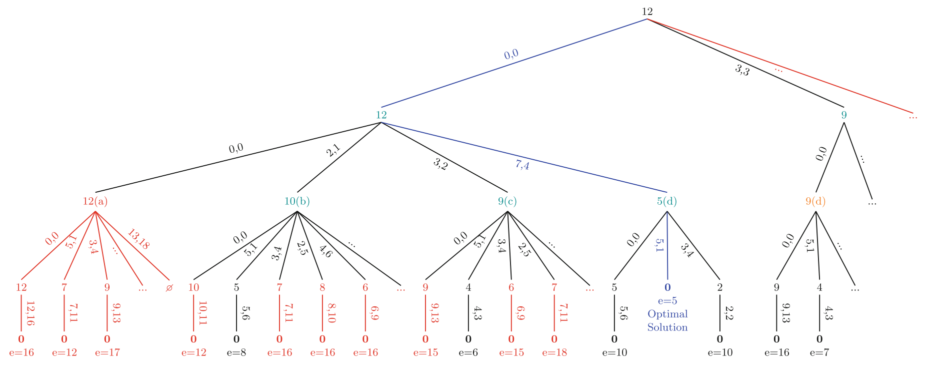


Fig. 3. Applying naive approach to examine all combinations and select a workload distribution with the minimum dynamic energy consumption.

internal node at level L_1 (root's children), determines the remaining workload to be distributed between processors $\{P_1, P_2, P_3\}$.

Similarly, each child of the root in the next level L_1 turns into a root of a sub-tree, which is a solution tree to solve HEOPT for the remaining workload between three processors $\{P_1, P_2, P_3\}$. Each edge, which connects the root and its child, is labelled by the problem size assigned to P_0 and its energy consumption.

In Fig. 3, the leaf node at level L_1 labelled by 0 represents a solution leaf. Generally, any leaf node labelled by 0 illustrates one of the possible solutions, where its dynamic energy consumption is calculated as the summation of the consumed energies labelling the edges in the path connecting the root and the solution leaf. *No-solution* leaves are labelled by $\varnothing$.

In this example, the distribution $\{(0,0),(7,4),(5,1),(0,0)\}$, highlighted in blue, with the consumed dynamic energy of 5, represents the optimal solution.

The cost of this naive algorithm is exponential. HEOPTA utilizes two bounding criteria, *energy threshold* and *size threshold*, and saving the intermediate solutions to find optimal solutions in a polynomial complexity of $O(m^3 \times p^3)$.

The *energy threshold*, represented by ε, is the dynamic energy consumption of load-equal distribution, allocating each processor the same workload of size $\frac{n}{p}$ (assuming n is divisible by p). HEOPTA will not examine data points with the dynamic energy consumption greater than or equal to the energy threshold.

The *size threshold* assigns each level of the tree a threshold, $\sigma_i, i \in \{0, \ldots, p-1\}$, which represents the maximum workload that can be executed in parallel on processors $\{P_i, \cdots, P_{p-1}\}$ so that the dynamic energy consumption by every processor $\{P_i, \cdots, P_{p-1}\}$ is less than ε.

HEOPTA explores solution trees in the left-to-right depth-first order as shown in Fig. 3. Before exploring a branch, the branch is checked against two upper estimated bounds, *energy threshold* and *size threshold*, and is discarded if it cannot result in a better solution than the best one found so far. All subtrees, not explored by applying the bounding criteria, are highlighted in red in Fig. 3. We call this key optimization operation *Cut*.

When a solution is found, the following operations are performed: (i) The energy threshold ε is updated, (ii) If ε decreases, the vector σ of size thresholds is updated, and (iii) The solution is saved in the memory. Green nodes in the tree highlight ones whose solutions are saved. We call this key operation, *Save*. Before exploring a node, HEOPTA read the memory to retrieve its solution (if it have already been saved). This key operation is called ReadMemory. The solution of the orange node in the tree is retrieved from the memory.

In summary, HEOPTA uses three key operations, *Cut*, *Save*, and ReadMemory, to find the optimal solutions. In supplemental available online in [11], we elucidate using an example how these key operations reduce the search space of solutions. The pseudocode of HEOPTA, its correctness and complexity proofs are also presented in the supplemental in [11].

5 Device-Level Dynamic Energy Decomposition in Heterogeneous Hybrid Platforms

We describe our practical approach here to construct the discrete dynamic energy profiles of the abstract processors in a hybrid heterogeneous server. The method is based purely on system level measurements. The approach comprises of two main steps. The first step is the identification or grouping of the computing elements satisfying properties that allow measurement of their energy consumptions to sufficient accuracy. We call these groups as *abstract processors*. The second step is the construction of the dynamic energy models of the abstract processors where the principal goal apart from minimizing the time taken for model construction is to maximize the accuracy of measurements.

5.1 Grouping of Computing Elements

We group individual computing elements executing an application together in such a way that we can accurately measure the energy consumption of the group. We call these groups *abstract processors*. We consider two properties essential to composing the groups:

– *Completeness:* An abstract processor must contain only those computing elements which execute the given application kernel.
– *Loose coupling:* Abstract processors do not interfere with each other during the application. That is, the dynamic energy consumption of one abstract processor is not affected by the activities of other abstract processor.

Based on this grouping approach, we hypothesize that the total dynamic energy consumption during an application execution will equal the sum of energies consumed by all the abstract processors. So, if E_T is the total dynamic energy consumption of the system incorporating p abstract processors $\{AP_1, \cdots, AP_p\}$, then $E_T = \sum_{i=1}^{p} E_T(AP_i)$, where $E_T(AP_i)$ is the dynamic energy consumption of the abstract processor AP_i. We call this our *additive* hypothesis.

5.2 Energy Models of Abstract Processors

We describe here the second main step of our approach, which is to build the dynamic energy models of the p abstract processors. We represent the dynamic energy model of an abstract processor by a discrete function composed of a set of points of cardinality m. The total number of experiments available to build the dynamic energy models is $(2^p - 1) \times m$. Consider, for example, three abstract processors {A,B,C}. {A,B,C, {AB,C}, {A,BC}, {AC,B}, ABC}. The category {AB,C} represents parallel execution of application kernels on A and B followed by application kernel execution on C. For each workload size, the total dynamic energy consumption is obtained from the system-level measurement for this combined execution of kernels. The categories {AB,C} and {BA,C} are considered indistinguishable. There are m experiments in each category. The goal is to construct the dynamic energy models of the three abstract processors {A,B,C} from the experimental points to sufficient accuracy. We reduce the number of experiments to $p \times m$ by employing our additive hypothesis.

6 Experimental Results

We employ two connected heterogeneous multi-accelerator NUMA nodes, HCLServer1 (Table 1) and HCLServer2 (Table 2). HCLServer1 is modelled by three abstract processors, CPU_1, GPU_1 and PHI_1, as described earlier. HCLServer2 is modelled by two abstract processors, CPU_2 and GPU_2.

We employ two popular data-parallel applications, matrix-matrix multiplication (DGEMM) and 2D fast Fourier transform (2D-FFT). Each application executing on the servers in parallel consists of five kernels, one kernel per computational device. Figure 1 shows discrete dynamic energy functions for the five abstract processors for DGEMM and 2D-FFT. For the DGEMM application, workload sizes range from 64×10112 to 28800×10112 with a step size of 64 for the first dimension m. For the 2D-FFT application, workload sizes range from 1024×51200 to 10000×51200 with a step size of 16 for the first dimension m.

For measuring dynamic energy consumption, each node is facilitated with one WattsUp Pro power meter which sits between the wall A/C outlets and the input power sockets of the node. Each power meter captures the total power consumption of one node. We use *HCLWattsUp* API [8], which gathers the readings from the power meter to determine the dynamic energy consumption during the execution of an application. HCLWattsUp has no extra overhead and therefore does not influence the energy consumption of the application execution. Fans are significant contributors to energy consumption. To rule out the contribution of fans in dynamic energy consumption, we set the fans at full speed before executing an application.

For each data point in the functions, the experiments are repeated until sample means of all the five kernels executing on the abstract processors fall in the confidence interval of 95%, and a precision of 0.1 (10%) is achieved.

Our approach on how to instrument computational kernels in a hybrid application and measure their execution times and dynamic energies is explained in detail in [11]. We also present our analysis of the accuracy of the additive approach to constructing discrete dynamic energy profiles in [11].

While the proposed method is rather expensive and requires significant time to build the energy profiles, the alternative approaches, namely, on-chip power sensors, such as Intel RAPL [7], Nvidia NVML [17], or AMD APM [4], and software models using performance counters as predictor variables, are still too inaccurate for the use in application-level optimization for energy [6,18].

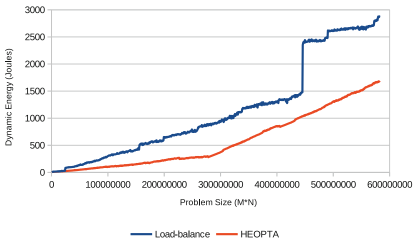
6.1 Analysing HEOPTA

HEOPTA is analysed using two sets of experiments. For the first set, we compare the dynamic energy consumption of solutions determined by HEOPTA with the dynamic energy of load-balanced solutions. Load-balanced solutions are workload distributions with equal execution times for each abstract processor. The number of active processors in a solution (those assigned non-zero workload size) may be less than the total number of available processors. The dynamic energy saving against load-balancing algorithm is obtained as follows: $Energy\_Saving_{balance}(\%) = \frac{e_{balance} - e_{heopta}}{e_{heopta}} \times 100$, where $e_{balance}$ and e_{heopta} are the dynamic energy consumptions of solutions determined by load-balancing and HEOPTA algorithms.

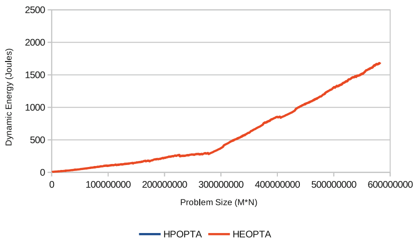
For the second set, we examine the interplay between dynamic energy optimization and performance optimization using the workload distribution determined by HPOPTA. HPOPTA [10] is a data-partitioning algorithm for optimization of data-parallel applications on heterogeneous HPC platforms for performance. The energy saving of HEOPTA against HPOPTA is obtained as follows: $Energy\_Saving_{hpopta}(\%) = \frac{e_{hpopta} - e_{heopta}}{e_{heopta}} \times 100$, where e_{hpopta} represents the dynamic energy consumption of the solution determined by HPOPTA. The inputs to HPOPTA are discrete speed (or performance) functions.

The experimental dataset for DGEMM contains the workload sizes, $\{64 \times 10112, 128 \times 10112, \cdots, 57600 \times 10112\}$. The minimum, average, and maximum reductions in the dynamic energy consumption of HEOPTA against load-balancing algorithm, $Energy\_Saving_{balance}$, are 0%, 130%, and 257%. Zero percentage improvement represents the same workload distribution is determined by HEOPTA and load-balancing algorithm. These values for $Energy\_Saving_{hpopta}$ are 0%, 145%, and 314%. Figure 4 compares HEOPTA against the dynamic energy consumption of solutions determined by load-balancing and HPOPTA. Performance optimization increases dynamic energy consumption by an average of 145%.

The experimental data set for 2D-FFT includes workload sizes, $\{1024 \times 51200, 1040 \times 51200, \cdots, 20000 \times 51200\}$. The minimum, average, and maximum dynamic energy reductions of HEOPTA against load-balancing algorithm, $Energy\_Saving_{balance}$, are 0%, 44%, and 105%. The minimum, average, and maximum of $Energy\_Saving_{HPOPTA}$ are 0%, 32%, and 77%. Figure 5 compares HEOPTA against the dynamic energy consumption of solutions determined by load-balancing and HPOPTA. Optimization for performance increases dynamic energy consumption by an average of 32%.

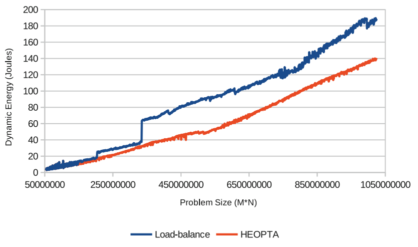


(a) HEOPTA vs load-balancing

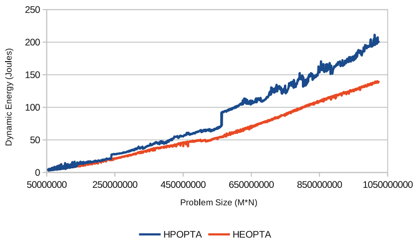


(b) HEOPTA vs HPOPTA

Fig. 4. Dynamic energy consumption of DGEMM executed using HEOPTA in comparison with (a) Load-balanced solutions (b) HPOPTA.



(a) HEOPTA vs load-balancing



(b) HEOPTA vs HPOPTA

Fig. 5. Dynamic energy consumption of the 2D-FFT application executed using HEOPTA in comparison with (a) Load-balanced solutions, (b) HPOPTA.

We conclude that HEOPTA demonstrates considerable improvements in average and maximum dynamic energy consumptions for the two applications in comparison with the load-balancing and HPOPTA algorithms. Performance optimization also increases dynamic energy consumption for both applications.

7 Conclusion

Modern heterogeneous HPC platforms feature tight integration of multicore CPUs with accelerators, which resulted in inherent complexities. One visible manifestation of these complexities is a complex functional relationship between energy consumption and workload size of applications executing on these platforms thereby providing an opportunity for application-level energy optimization through workload distribution as a decision variable.

We proposed HEOPTA that determines optimal workload distributions minimizing the dynamic energy consumption of data-parallel applications running on heterogeneous HPC platforms. We showed that the load-imbalanced solutions provided by the algorithm achieve significant dynamic energy reductions compared to the load balanced solutions. As future work, we will study the impact of dynamic energy optimization on performance.

The software implementation for HEOPTA is available at [11].

References

1. Basmadjian, R., Ali, N., Niedermeier, F., de Meer, H., Giuliani, G.: A methodology to predict the power consumption of servers in data centres. In: 2nd International Conference on Energy-Efficient Computing and Networking. ACM (2011)
2. Chakrabarti, A., Parthasarathy, S., Stewart, C.: A pareto framework for data analytics on heterogeneous systems: implications for green energy usage and performance. In: 46th International Conference on Parallel Processing (ICPP), pp. 533–542. IEEE (2017)
3. Choi, J., Dukhan, M., Liu, X., Vuduc, R.: Algorithmic time, energy, and power on candidate HPC compute building blocks. In: IEEE 28th International Parallel and Distributed Processing Symposium, pp. 447–457. IEEE (2014)
4. Devices, A.M.: Bios and kernel developer's guide (BKDG) for AMD family 15h models 00h–0Fh processors (2012). https://www.amd.com/system/files/TechDocs/42301_15h_Mod_00h-0Fh_BKDG.pdf
5. Drozdowski, M., Marszalkowski, J.M., Marszalkowski, J.: Energy trade-offs analysis using equal-energy maps. Future Gener. Comput. Syst. **36**, 311–321 (2014)
6. Fahad, M., Shahid, A., Manumachu, R.R., Lastovetsky, A.: A comparative study of methods for measurement of energy of computing. Energies **12**(11), 2204 (2019)
7. Gough, C., Steiner, I., Saunders, W.: Energy Efficient Servers: Blueprints for Data Center Optimization. Apress, New York (2015)
8. HCL: HCLWattsUp: API for power and energy measurements using WattsUp Pro Meter (2016). https://csgitlab.ucd.ie/ucd-hcl/hclwattsup
9. Hsu, J.: Three paths to exascale supercomputing. IEEE Spectr. **53**(1), 14–15 (2016)
10. Khaleghzadeh, H., Manumachu, R.R., Lastovetsky, A.: A novel data-partitioning algorithm for performance optimization of data-parallel applications on heterogeneous HPC platforms. IEEE Trans. Parallel Distrib. Syst. **29**(10), 2176–2190 (2018)
11. Khaleghzadeh, H., Reddy, R., Lastovetsky, A.: HEOPTA: heterogeneous model-based data partitioning algorithm for optimization of data-parallel applications for dynamic energy (2019). https://csgitlab.ucd.ie/HKhaleghzadeh/heopt
12. Lang, J., Rünger, G.: An execution time and energy model for an energy-aware execution of a conjugate gradient method with CPU/GPU collaboration. J. Parallel Distrib. Comput. **74**(9), 2884–2897 (2014)
13. Lastovetsky, A., Reddy, R.: New model-based methods and algorithms for performance and energy optimization of data parallel applications on homogeneous multicore clusters. IEEE Trans. Parallel Distrib. Syst. **28**(4), 1119–1133 (2017)
14. Manumachu, R.R., Lastovetsky, A.: Bi-objective optimization of data-parallel applications on homogeneous multicore clusters for performance and energy. IEEE Trans. Comput. **67**(2), 160–177 (2018)
15. Marszałkowski, J.M., Drozdowski, M., Marszałkowski, J.: Time and energy performance of parallel systems with hierarchical memory. J. Grid Comput. **14**(1), 153–170 (2015). https://doi.org/10.1007/s10723-015-9345-8
16. Nagasaka, H., Maruyama, N., Nukada, A., Endo, T., Matsuoka, S.: Statistical power modeling of GPU kernels using performance counters. In: International Green Computing Conference and Workshops (IGCC). IEEE (2010)
17. Nvidia: Nvidia management library: NVML reference manual, October 2018. https://docs.nvidia.com/pdf/NVML_API_Reference_Guide.pdf
18. O'Brien, K., Pietri, I., Reddy, R., Lastovetsky, A., Sakellariou, R.: A survey of power and energy predictive models in HPC systems and applications. ACM Comput. Surv. **50**(3), 37 (2017)

19. Shao, Y.S., Brooks, D.: Energy characterization and instruction-level energy model of Intel's Xeon Phi processor. In: Proceedings of the 2013 International Symposium on Low Power Electronics and Design, ISLPED 2013. IEEE Press (2013)
20. Song, S., Su, C., Rountree, B., Cameron, K.W.: A simplified and accurate model of power-performance efficiency on emergent GPU architectures. In: 27th IEEE International Parallel and Distributed Processing Symposium (IPDPS), pp. 673–686. IEEE Computer Society (2013)
21. Zhong, Z., Rychkov, V., Lastovetsky, A.: Data partitioning on multicore and multi-GPU platforms using functional performance models. IEEE Trans. Comput. **64**(9), 2506–2518 (2015)