International Innovative Research Journal of Engineering and Technology
ISSN: 2456-1983 Vol: 5, Issue No: 1, September 2019

An ILP Structure for Energy Optimization of Non-Preemptive Real Time Tasks on Multiprocessors

Shahirah Mohamed Hatim[1], **Habee Bullah bin Affandy**[2]
[1]Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA, Perak Branch, Tapah Campus, 35400 Tapah Road, Perak. Malaysia
[2]School of Human Development and Technocommunication, Universiti Malaysia Perlis, Perlis, Malaysia.

Abstract: Among this article, we approach the issue of scheduling a set of regular non-preemptive real-time functions on a heterogeneous multiprocessor platform by taking into consideration both the views of feasibility and energy consciousness. We suppose processors that are allowed for dynamic voltage frequency scaling. The goal is to calculate a viable timetable leading to the optimization of target energy requirements. In such a job, we consider some frequently used energy optimization requirements such as general energy consumption, balancing power consumption among processors, concurrent optimization of complete energy consumption as well as energy balancing among processors, maximum energy consumption. We describe a computational method depending on Integer Linear Programming (ILP) to determine the optimum solution.

Keywords: Energy-Aware System, Dynamic Voltage Scheduling, Heterogeneous Multiprocessor Scheduling.

1. INTRODUCTION

In the current situation, energy consumption in the cyber-physical system is one of the primary issues. With the proliferation of Internet-of-Things allowed devices to be run in low-power mode, more rational handling of energy use is required. Also, these devices are performed using energy-limited battery power. Thus, one of the major problems was to develop low-power IoT or cyber-physical systems. It is of utmost importance to design cyber-physical systems efficiently which consume less energy and improve general system efficiency. Real-time limitations that are common for critical safety systems add another dimension to the issue. Being a non-trivial and difficult one creates the issue. Therefore, in cyber-physical systems, we delve into energy-efficient scheduling of real-time activities. A real-time system typically comprises a set of activities to be planned on a set of computing resources. The duties can vary in terms of periodicity, the strategy of preemption, power consumption, length of execution, etc.

The conduct of a job can be influenced by several other system parameters such as voltage, frequency, etc. An interesting issue to address is the energy-efficient design of real-time devices in the context of non-preemptive assignments. The obstacles to this issue stem from the two critical problems. One of them was the feasibility of scheduling assignments considering their corresponding deadlines. Another is to improve the mathematical computations ' timetable of execution for improved energy efficiency. It is known that the issue of schedulability for non-preemptive assignments is intractable. Many safety-critical systems have tough deadlines for the assignments. Interestingly, for the following reasons, many security-critical systems prefer the non-preemptive scheduling policy.

In such a system, a job remains to also be performed until it is complete. Furthermore, there is no change of context on the processor. Therefore, no additional memory will be needed to store context-related data. Whether the functions had some frameworks, then the handling of mutual exclusion will not require special attention. Also, non-preventive scheduling has the benefit of program location; thus, timing behavior prediction can be produced more correctly. Compared to preemptive counterpart, the main problem with the non-preemptive planning system is the lack of a simple technique for schedulability assessment. Schedulability of preventive assignments can be easily determined by the processes calculating the complete use of the resource.

Analysis of the schedulability of non-preemptive assignments is much less explored in the literature. In addition to scheduling, optimizing the tasks ' power usage is also a stimulating issue.

Because computing resources are presumed to be activated by DVFS, a task's execution length differs based on the voltage level. Therefore, it is necessary to relook the schedulability of the duties. An embedded method is therefore needed to simultaneously analyze schedulability and energy optimization.

With a multidimensional attempt, the low-power system design issues have been discussed over the past decade. Reducing a system's energy consumption includes distinct parts like CPU, memory system, and I / O subsystem with necessarily distinct elements. Although the processor's energy reduction is an apparent goal, the early accounts showed that complete power usage is 18%-30% owing to the CPU alone.

But, the latest research demonstrates that owing to a rise in processor workloads, the proportion can exceed 50 percent. Dynamic voltage/frequency scaling (DVFS) framework has been widely used to reduce the CPU power consumption, which involves dynamically adjusting the voltage and frequency.

A significant quantity of job has been done on selecting dynamic voltage to minimize power. A DVFS method has been suggested in Ref.[4] for single processor devices that can alter the supply voltage over a constant spectrum.

For the issue of discrete voltage choice, an integer linear programming (ILP) formulation has been proposed5. A linear programming (LP) solution with uniform and non-uniform switched capacitance described in Ref.[6] for the discrete voltage selection problem.

Because prior research gives the notion that in polynomial time the issue can be solved optimally, it has been demonstrated that the issue of discrete voltage choice is indeed NP-hard. For a set job ordering without considering communication time and energy, an ILP formulation for voltage scaling provided. An ILP formulation was suggested in Ref.[8] for the choice of constant supply voltage in distributed systems. It often solved by an approximation the issue of a discrete choice of voltage. Some research has been done to address the single-processor variable voltage scaling with autonomous functions. An ideal planning method is defined on a single processor for autonomous assignments with variable velocity. Viable non-preemptive scheduling of duties has been discussed on a single processor and multiprocessor, minimizing the complete energy used. A heuristic low-energy schedule on a single processor core with variable supply voltage suggested for non-preemptive autonomous assignments

2. LITERATURE REVIEW

In the literature, for heterogeneous distributed systems, heuristics were mostly suggested for dynamic voltage choice. A list of heuristic scheduling is used to minimize power. For dependent assignments on a distributed scheme comprising varying supply voltage processors, a unique priority feature for non-preemptive scheduling on processors with varying supply voltage scheduling method has been suggested. A priority system based on average energy consumption is provided which, whenever an infeasible timetable is discovered and tasks are rescheduled, dynamically improved the priorities of the assignments. A scheduling algorithm for the dependent assignments on the heterogeneous processors has been provided in such a way as to minimize complete power usage while meeting the task precedence limitations and deadline limitations.

To minimize the peak power usage on heterogeneous processors for frame-based and periodic real-time tasks by scheduling the sleep cycles for each active processor and proposing the concept of a sufficient peak power consumption test for task feasibility. It is regarded as the optimization of maximum energy load and system reliability.

In this job, we focus on the non-preemptive regular real-time task scheduling in such a way that the energy consumption efficiency of the system is enhanced. The role of power optimization may differ from implementation to implementation. The general power consumption is minimized in many practical apps, and this is a well-known criterion for optimization. It is not only the complete energy but also the standardized energy consumption across various computing resources that is crucial in a multiprocessor setting.

The peak-to-average power usage ratio is optimized in many devices. This work discusses an integer linear programming-based energy optimization framework where a set of limitations to describe acceptable system conduct will be established. It is necessary to select the suitable objective function of the designer depending on the requirement. Below are a few commonly used objective features.

Minimizing complete power usage in a scheme is often aimed. Overall energy consumption could be calculated for a multiprocessor scheme through summarizing the power consumption in each processor. While such an important criterion decreases general energy consumption, it may not guarantee equilibrium between the available processors in energy consumption.

Balancing energy consumption across processors in a multiprocessor setting is also a crucial objective criterion. If the energy consumption balance is not met, some processors may be considerably heated compared to other processors. Excessive heat at a tiny chip place can lead to processor malfunction.

The earlier goal is to guarantee energy consumption uniformity among processors. Complete energy consumption can be improved to obtain uniformity. Therefore, optimizing complete energy consumption and keeping energy consumption equilibrium across processors will be nice.

Minimizing the processors ' maximum power usage is another important criterion for the design of the scheme. As this restricts a processor's maximum energy consumption, complete power usage is not anticipated to be very big.

The following is a short overview of this article. We present the model of the scheme and formulate the issue in Section 2.To achieve the optimum solution in Section 3, we provide an integer linear programming formulation. Section 4 presents the outcomes of the experimental study. Finally, in Section 5, the conclusion is provided.

3. RESEARCH AND METHODOLOGY

We introduce the system model in this chapter and formulate the issue to be resolved.

3.1 System Model

We find a set of regular real-time functions that are non-preemptive,$\mathbb{T} = \{T_1, \dots \dots .. T_n\}$. A task is characterized by $\langle e_x, p_x \rangle$. Where e_x, and $p_x (\geq e_x)$. Represents the worst-case performance time and work time T_x, respectively. We assume a multiprocessor heterogeneous system consisting of $\mathbb{P} = \{P_1, \dots \dots .. P_n\}$ m processors.
With a discrete set of $\mathfrak{l}$ frequency levels, each $P_i \in \mathbb{P}$ processor can run $F_i = \{F_i^1, \dots \dots, F_i^{\mathfrak{l}}\} | F_i^j < F_i^{j+1}$. Thus, $F_i^{\mathfrak{l}}$ and F_i^1represent the maximum and minimum frequency atP_i, respectively. Without loss of generality, we assume the frequency values will be standardized so that the maximum frequency at any processor will be$F_i^j \leq 1.0$
We suppose that $e_x$on a normalized frequency $F_i^{\mathfrak{l}} = 1.0$. is the execution moment. We also suppose that the assignments have an implicit time limit.
Let's regard T_{xy} as the yth example of T_x assignment, i.e. the yth activation of T_x published at $(y-1)P_x$. with yp_x deadline. Thus, $(y-1)P_x$and yp_x, respectively, represent T_{xy}'s release time and deadline.The hyper-period$\rho$is described as $\rho = \mathrm{lcm}\{P_1, \dots \dots .. P_n\}$and the number of $T_x$cases in the hyper-period are described as $\omega_x = \rho / P_x$. Let, $f_{xy} \in F_i$be the frequency of execution of T_{xy}. Effective T_{xy} execution time at f_{xy} is, therefore, $e_x = \lceil e_x / f_{xy} \rceil$.
In CMOS technology, power consumption is largely dominated by dynamic power dissipation$P$, which is provided as

$$P \propto V_{dd}^2 \times f \qquad (1)$$

Where f is the clock frequency and V_{dd} is the supply voltage with a linear connection to the processing frequency f. Consequently, Eq. (1) could be rescheduled as

$$P \propto f^3 \qquad (2)$$

When voltage/frequency scaling is performed using the DVFS method, the frequency of the processor can be adjusted within the range between F_i^1 and $F_i^{\mathfrak{l}}$. The power consumption for the T_{xy} work is therefore calculated as

$$E_{xy} = f_{xy}^2 \times e_x \qquad (3)$$

3.2 Problem Statement

We are provided a set of non-preemptive$\mathbb{T} = T_x$assignments, where $T_x = \langle e_x, p_x \rangle$, a set of $\mathbb{P} = \{P_i\}$ processors where the processors are allowed with DVFS and have the amount of frequency as$F_i = \{F_i^1, \dots, F_i^{\mathfrak{l}}\}$. Our goal is to plan duties in an energy-optimised manner on the accessible processors as indicated by the cost function. We need to take care of the following duties while scheduling the assignments. Our goal is to plan duties in an energy-optimized manner on the accessible processors as indicated by the cost function. We have to take care of the following limitations while scheduling the duties.
C1: At one moment, only one task can be performed by a processor.
C2: A task example must run on a processor.
C3: A task instance can only run on a single processor and can not be preempted.
C4: An instance of a task shall be scheduled for execution on a single processor only before the time limit implicitly specified by the time limit.
C5: A task example shall only run at a single frequency. Multiple task cases may have distinct frequencies.
Regardless of the optimization function, the above limitations are to be met. We consider the four optimization criteria that are most commonly used and these are as follows. We use $E_i$to indicate the $P_i$processor's energy consumption.
First, we declare minimizing complete energy consumption as one of the important measures for comparing system efficiency. This can be conveyed in mathematical terms as

$$\mathbb{E}_{\text{tot}} = \sum_{i=1}^{m} E_i \qquad (4)$$

Then, we're trying to balance the available processors' power usage. As a balance measure, we use the amount of the pair wise absolute energy consumption difference on the processors. We can mathematically convey this as follows:

$$\mathbb{E}_{\text{balance}} = \sum_{i} \sum_{i'} |E_i - E_{i'}|, \quad \forall i, i', i \neq i' \qquad (5)$$

Since the last objective function attempts to balance the energy consumption among the processors, the complete energy consumption can significantly improve.
To prevent such a scenario, we are formulating a distinct objective function that attempts to take care of complete energy consumption as well as maintaining a balance on the processors' power usage. For optimization, the following mathematical expression is used.

$$\mathbb{E}_{\text{combined}} = \sum_{i=1}^{m} E_i + \sum_{i} \sum_{i'} |E_i - E_{i'}|, \quad \forall i, i', i \neq i' \qquad (6)$$

4. ILP FORMULATION FOR THE PROBLEM

There are several methods such as branch-and-bound, A ubiquitous search, limited fulfillment, etc. to find an ideal solution to the above-mentioned issues. In this job, we choose to model the issue using an approach to integer linear programming (ILP), a limited version of the issue of limited optimization.

The reason for leaving the technique based on the ILP is to follow. A comparable framework can be used to implement many variations of suggested objective features. Moreover, under many restricted environments, such a model offers designers an easy way of thinking about the system's conduct.

A set of decision variables must be defined and the constraints expressed using those variables to formulate the problem using ILP. It will also describe the objective function using the same set of variables.

The solver will assign the values to the factors of the choice to minimize or maximize the optimal function based on the situation.

Predicated on the limitations listed in the last chapter, we can conclude that execution on any processor (C1) can begin at a maximum of one example of a job. Let T_x launches its execution with the jth frequency at moment t on the P_i processor. Then at the moment t on$P_i$no other job can start execution. This limitation can be described as:

$$\mathrm{C1}: \sum_{x}\sum_{j} v_{xijt} \le 1, \quad \forall i, t \qquad (8)$$

Each example of each assignment must be executed on some processor during its fixed period (C2) interval. We can use the following to convey such restrictions.

$$\mathrm{C2}: \sum_{i}\sum_{j}\sum_{t=(y-1)p_x}^{yp_x} v_{xijt} = 1, \quad \forall x, y \in \{1, \ldots, \rho/p_x\} \quad (9)$$

Therefore, as we assumed the non-preemptive task set, once a task starts executing on a processor, the same processor will not be available for the other tasks until the executing task is complete (C3). This constraint can be articulated in the following manner mathematically.

If a job begins executing on P_i at the moment t then no other job can begin execution on P_i till $t + e_{xij}$ where e_{xij} represents T_{xy}'s efficient execution time executing on the P_i processor at frequency F_i^j.

$$\mathrm{C3}: v_{xijt} + \sum_{x'}\sum_{j'=1}^{F_i}\sum_{t'=t}^{t+e_{xij}} v_{x'ij't'} \le 1, \quad \forall x, x' \in [1, n],$$

$$x \ne x', i, j, t \in t_{xy} \; (10)$$

Perhaps we can express the constraint that each example of each assignment must fulfill its deadline as indicated by its time limit (C4). Let's consider that T_{xy} begins with F_i^j frequency at the moment t on the P_i processor. So at the time $t + e_{xij}$, the task will finish its execution, and it must be less than theyp_x.

We constitute the following limitation.

$$\mathrm{C4}: t \times v_{xijt} + e_{xij} \le \left\lceil \frac{t}{p_x} \right\rceil \times p_x \qquad (11)$$

We don't need to explicitly convey the C5 restriction. The way we selected the choice variables is to execute a single frequency instance. We identify the energy consumption of each example of each assignment before proceeding to describe the objective features. The energy consumption to perform the $T_x$task yth example (running on the $P_i$processor) is

$$E_{xyi} = \sum_{j}\sum_{t \in t_{xy}} v_{xijt} \times \left((F_i^j)^2 \times e_{xij} \right), \quad \forall x, i \qquad (12)$$

The total energy consumption of the Pi processor is calculated as

$$E_i = \sum_{x}\sum_{y=1}^{\rho/p_x} E_{xyi} \quad \forall i \qquad (13)$$

So we're talking about the objective tasks. We can use equations for the first three situations as presented in Section 2. (4)–(6) as our function of purpose. However, we need to introduce an additional variable for the fourth one (Eq. (7)), which is the processor's minimization of maximum energy consumption. Use an additional variable to indicate the processor's maximum energy consumption. We can, therefore, add the following restrictions.

$$A1: E_i \le \varepsilon \quad \forall i \qquad (14)$$

It is possible to express the objective function as

Subject to limitations as outlined in equations (8)–(14)

5. RESULTS

We introduced the suggested ILP-based model and used the IBM cplex tool to solve the optimization problem. We conducted comprehensive experiments on randomly produced test instances and compared total energy consumption, power load difference between processors, maximum energy load, etc.

All experiments were conducted on Intel computer with a clock speed of 128 GB RAM, 3.30 GHz. We select the parameters of the specified range-period [10–50], the worst-case implementation time [1–10], and the standardized frequency set for each processor [0.1–1.0].

First, we describe the outcomes of various criteria for optimization. We vary the number of tasks in the range 6 to 20 for our experimentation, and we generate 100 test cases randomly for each category. For this experiment, the amount of processors considered is 10. In Table 1, we demonstrated the general energy consumption and peak energy load using various cost optimization functions.

Column I indicate the number of assignments between 6 and 20. Column-(II – V) indicates the average optimal calculated value for various objective tasks. It can be noted that with the number of assignments the value improves.

Column-(VI – IX) and Column-(X – XIII) show the complete power usage and maximum power load by distinct optimization criteria. It can be noted that complete power consumption using $\mathbb{E}_{combined}$ is 2 percent –30 percent difference from which provides the system's optimum power usage $\mathbb{E}_{tot}$.

However, the combined peak energy load of $\mathbb{E}_{combined}$ is lower than that of $\mathbb{E}_{tot}$. Total use of $\mathbb{E}_{balance}$ and $\mathbb{E}_{peak}$ energy is greater than $\mathbb{E}_{combined}$. Although $\mathbb{E}_{peak}$'s smallest minimum peak power load among others, $\mathbb{E}_{combined}$ also performs well in terms of peak power load.

From the table, it can be noted that $\mathbb{E}_{balance}$'s general energy consumption and peak energy load is relatively greater than other features of optimization.

We also provided the computing time and memory utilization in Table 2 to solve various objective functions. Column-(III–V) indicates the computing time in seconds, and Column-(VI-VIII) indicates the memory requirement in MB. It can be seen that $\mathbb{E}_{tot}$ and $\mathbb{E}_{peak}$ take much less time to calculate and complete within 1 minute. But, it takes more than 60 seconds to calculate both $\mathbb{E}_{combined}$ and $\mathbb{E}_{balance}$, and increases with the range of tasks as the lookup space rise with a set of variables (shown in Column-I).

From the table it can be seen that memory use improves considerably with the number of assignments; however, compared to other optimization criteria, $\mathbb{E}_{tot}$ needs less memory.

Depending on the observation, $\mathbb{E}_{tot}$ could conclude that if computing resources are restricted. On the other side, in a similar moment and memory consumption, $\mathbb{E}_{peak}$ provides excellent performance alternatives in terms of general energy and peak energy load.

Table 1. Comparison of objective functions

Objective Function				
Tasks	E_{tot}	$E_{balance}$	$E_{combined}$	E_{peak}
I	II	III	IV	V
6	6.04	32.53	50.98	1.18
8	6.41	34.97	53.76	3.24
10	7.48	38.26	57.62	3.74
14	9.09	45.2	65.31	5.12
18	11.43	49.31	68.58	5.62
20	14.56	52.07	71.46	6.05

Total Energy			
E_{tot}	$E_{balance}$	$E_{combined}$	E_{peak}
VI	VII	VIII	IX
6.04	6.67	6.17	6.33
6.41	7.23	6.98	7.08
7.48	9.02	9.24	9.6
9.09	11.01	10.26	11.23
11.43	15.53	12.32	14.6
14.56	20.6	18.6	19.6

Peak Energy load			
E_{tot}	$E_{balance}$	$E_{combined}$	E_{peak}
X	XI	XII	XIII
3.1	3.9	1.44	1.18
3.94	5.64	4.46	3.24
4.76	5.52	4.78	3.74
5.91	7.36	6.48	5.12
6.82	6.58	6.14	5.62
6.79	7.15	6.56	6.05

Table 2. Comparison of computation time and memory usages

No. of Variables		Computation Time (sec)				Memory Usages (MB)			
Tasks	$x10^3$	E_{tot}	$E_{balance}$	$E_{combined}$	E_{peak}	E_{tot}	$E_{balance}$	$E_{combined}$	E_{peak}
I	II	III	IV	V	VI	VII	VIII	IX	X
6	4.6	0.41	49	80	0.72	0.146	0.598	1035	0.236
8	5.2	0.65	72	108	4.32	0.831	1436	1735	1075
10	8.1	1.84	91	115	6.52	1088	1943	2434	1685
14	11.3	3.92	135	172	12.54	2432	4879	6778	3304
18	14.5	7.23	192	221	25.67	6738	9477	11368	7793
20	17.4	11.45	238	270	32.45	8324	11486	16744	9876

6. CONCLUSION

In this job, we discussed the issue of scheduling in a heterogeneous multiprocessor setting a set of non-preemptive regular real-time functions to optimize the energy requirements. We provided four distinct criteria for optimization. To fix the issue optimally, the issue is formulated in ILP. We look at total energy consumption and the maximum energy consumption by the various objective criteria. To solve the issue optimally, we also compare the computation time and memory requirement.

REFERENCES

[1] J. R. Lorch, A Complete Picture of the Energy Consumption of a Portable Computer. PhD thesis, Masters thesis, Department of Computer Science, University of California (1995).

[2] J. Pouwelse, K. Langendoen, and H. Sips, Dynamic voltage scaling on a low-power microprocessor, Proceedings of the 7th Annual International Conference on Mobile Computing and Networking, ACM, (2001), pp. 251–259.

[3] H. Zeng, C. S. Ellis, A. R. Lebeck, and A. Vahdat, Ecosystem: Managing energy as a first class operating system resource. ACM SIGOPS Operating Systems Review 36, 123 (2002).

[4] F. Yao, A. Demers, and S. Shenker, A scheduling model for reduced CPU energy, Proceedings of IEEE 36th Annual Foundations of Computer Science, IEEE (1995), pp. 374–382.

[5] T. Ishihara and H. Yasuura, Voltage scheduling problem for dynamically variable voltage processors, Proceedings of the 1998 International Symposium on Low Power Electronics and Design, ACM (1998), pp. 197–202.

[6] W.-C. Kwon and T. Kim, Optimal voltage allocation techniques for dynamically variable voltage processors. ACM Transactions on Embedded Computing Systems (TECS) 4, 211 (2005).

[7] A.Andrei, P. Eles, Z. Peng, M. T. Schmitz, and B. M. Al Hashimi, Energy optimization of multiprocessor systems on chip by voltage selection. IEEE Transactions on Very Large Scale Integration (VLSI) Systems 15, 262 (2007).

[8] Y. Zhang, X. S. Hu, and D. Z. Chen, Task scheduling and voltage selection for energy minimization, Proceedings of the 39th Annual Design Automation Conference, ACM (2002), pp. 183–188.

[9] V. Cohen-Addad, Z. Li, C. Mathieu, and I. Milis, Energy-efficient algorithms for non-preemptive speed-scaling, International Workshop on Approximation and Online Algorithms, Springer (2014), pp. 107–118.

[10] Hong, D. Kirovski, G. Qu, M. Potkonjak, and M. B. Srivastava, Power optimization of variable-voltage core-based systems. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 18, 1702 (1999).

[11] F. Gruian, Hard real-time scheduling for low-energy using stochastic data and DVS processors, ISLPED'01: Proceedings of the 2001 International Symposium on Low Power Electronics and Design (IEEE Cat. No. 01TH8581), IEEE (2001), pp. 46–51.

[12] F. Gruian and K. Kuchcinski, Lenes: Task scheduling for low-energy systems using variable supply voltage processors, Proceedings of the 2001 Asia and South Pacific Design Automation Conference, ACM (2001), pp. 449–455.

[13] B. Gorjiara, N. Bagherzadeh, and P. Chou, An efficient voltage scaling algorithm for complex socs with few number of voltage modes, Proceedings of the 2004 International Symposium on Low Power Electronics and Design, ACM (2004), pp. 381–386.

[14] Luo and N. K. Jha, Power-profile driven variable voltage scaling for heterogeneous distributed real-time embedded systems, 16th International Conference on VLSI Design, 2003. Proceedings, IEEE (2003), pp. 369–375.

[15] M. T. Schmitz and B. M. Al-Hashimi, Considering power variations of DVS processing elements for energy minimisation in distributed systems, Proceedings of the 14th International Symposium on Systems Synthesis, ACM (2001), pp. 250–255.

[16] K. Goh, B. Veeravalli, and S. Viswanathan, Design of fast and efficient energy-aware gradient-based scheduling algorithms heterogeneous embedded multiprocessor systems. IEEE Transactions on Parallel and Distributed Systems 20, 1 (2009).

[17] W. Munawar, H. Khdr, S. Pagani, M. Shafique, J.-J. Chen, and J. Henkel, Peak power management for scheduling real-time tasks on heterogeneous many-core systems, 2014 20th IEEE International Conference on Parallel and Distributed Systems (ICPADS), IEEE (2014), pp. 200–209.