Int. J. Nonlinear Anal. Appl. 13 (2022) 2, 2743–2750
ISSN: 2008-6822 (electronic)
http://dx.doi.org/10.22075/ijnaa.2022.6466

Task scheduling optimization based on heuristic algorithm for heterogeneous cloud computing platforms

Saeed Fatehi

Department of Computer Engineering, Babol Branch, Islamic Azad University, Babol, Iran

(Communicated by Javad Vahidi)

Abstract

In recent years, the issue of power consumption in parallel and distributed systems has attracted a great deal of attention. Regarding the ever-increasing development data and computing centers due to the contribution of cloud computing systems in such sectors, power consumption has always been of the concerns due to Carbon dioxide emissions and consequently the Negative impact on the environment. In recent years, the notion of power and also "Green Computing" has found a crucial spot in the tasks scheduling in cloud data centers. The clustering technique, as well as Dynamic Voltage and Frequency Scaling (DVFS) techniques, have focused on the reduction of the consumption of power particularly, and the optimization of the performance parameters. Concerning scheduling Directed Acyclic Graph (DAG) of a data center processors equipped with the technique of DVFS, this paper proposes a power and time aware algorithm called PATCDD, to apply the combination of the strategies for clustering along with the distribution of slack-time among the tasks of a cluster. The first phase studies the slack time for non-critical tasks of DAG, extends their execution time and reduces the energy consumption without increasing the task's execution time as a whole. The main idea of the proposed algorithm involves the achievement of a maximum reduction in power consumption in the second phase. To this end, the slack time is distributed among non-critical dependent tasks. Eventually, a set of data established for conducting the examinations and also different parameters of the constructed random DAG were assessed to identify the efficiency of our proposed algorithm.

Keywords: Cloud Computing, Dynamic Voltage and Frequency Scaling (DVFS), Slack time, Tasks Clustering
2020 MSC: 00A71, 03C30

1 Introduction

In recent years, the high price and cost of energy/power and a variety of environmental issues have forced the high performance computing sector to reconsider some of its old practices with an aim to create more sustainable HPC systems [2]. It has been estimated the total power and energy consumption in ICT industry over 868 billion kWh which accounts for 5.3% of the total electricity use across the globe in 2008, according to the reports [11]. The everincreasing technological advancement nowadays offers the expectation of up to four times the total power demand until 2025. Carbon emissions related to the energy uses may potentially increase the cooling cost. Any boosts in the usage of efficient computing systems will end in a very hot temperature caused by the increased generated power for fueling them. It has proven that some failures can be observed up to two times with the heat of rising of 10 C,

*Corresponding author
Email address: saeed.fatehi@yahoo.com (Saeed Fatehi)

Received: September 2021 *Accepted:* April 2022

that consequently affects the system reliability and availability which may result in severe damages [9]. As such, the power consumption for parallel task scheduling has heightened concerns in the last years. The specific approaches to the subject of energy conservation include dynamic power management (DPM) [16] and dynamic voltage frequency scaling (DVFS) already incorporated into many new processors [19]. The former usually aim to deactivate idle nodes due to the long durations of them. DPM approach shows no participation in short idle durations events due to a quick runtime between each task. Regarding the reduction of power consumption, however, the tasks receive a specified frequency in employing DVFS approach. Due to the power consumption and other efficient parameters, scheduling parallel tasks with limited prioritizing, i.e., directed acyclic graph is of concern in homogenous and heterogeneous computing environments such as cloud data centers. This issue is NP hard problems [15]. Clustering strategies, as well as DVFS, have confirmed to reduce energy consumption. The remainder of this paper is organized as follows. Section 2 introduces related work. In Section 3, we introduce the system model. In Section 4, we describe the proposed algorithm. Section 5 gives the experimental setup and simulation results. Finally, Section 6 presents the conclusion.

2 Related Work

In recent years there has been a significant research on task scheduling for embedded systems using various forms of DVFS enabled techniques. The main idea in most of the existing algorithms is to efficiently use processors' slack times to satisfy time requirements of all tasks such as execution times. The advancement in information and communication technology in recent years has caused to continuous growth cloud data centers and computing centers, accordingly increases in power consumption and negative impact on the environment through generation of greenhouse gases and excessive emission of CO2. In recent years, great efforts have been made on parallel task scheduling algorithms for distributed platforms such as clusters, girds and clouds, these studies achieved good results in power and performance improvements using DVFS techniques [2, 7], List scheduling [17], Clustering-based scheduling [2, 17] and heuristic algorithm [12, 1, 3]. This section gives a brief review about the various existing task scheduling algorithms which mainly consider the energy efficiency and optimizing performance parameters in distributed computing.

DVFS technique is one of the most efficient ways of providing significant power saving for processors through simultaneous minimization of frequency and supply voltage for slack time slot of tasks as well as communication and idle phases [4, 6].

The authors in [17] employed a power-aware scheduling heuristic algorithm called PALS and PATC to simultaneously reduce total execution time and power consumption for scheduling parallel tasks in a cluster computing through DVFS technique. After determining the critical path and non-critical paths and critical job and non-critical jobss, the proposed algorithm assigns jobs in the critical paths to processors with the highest voltage/frequency. Then, the slack time of each job is calculated in the non-critical paths, and the voltage/frequency of the assigned processors is scaled down to process the non-critical jobs. This strategy mitigates energy consumption without increasing makespan.

Another approach to scheduling tasks has been proposed to reduce power consumption using the DVFS technique [18]. This technique has been adopted to dynamically control the frequency and voltage of cloud computing servers. The scheduling algorithm takes into account the maximum job (Fmax) and minimum job (Fmin) frequencies given to each job and multiple server Si running at maximum Si (Fmax) and minimum Si (Fmin) frequencies. For specific jobs, the scheduling algorithm efficiently assigns proper servers that run between (Fmin, Fmax) to jobs according to requirements of job frequencies.

Juarez et al. [8] proposed a real-time dynamic scheduling method called Multi-heuristic Resource Allocation (MHRA) for efficient execution of task-based applications on a distributed computing platform of cloud computing. This served to mitigate energy consumption and makespan. This method involved a polynomial time algorithm combining a set of heuristic rules and resource allocation techniques. In order to balance the two-objective function, a weight factor was introduced α, by which the user can specify the significance of each objective.

Yikun Hu et al. [9] developed an EASLA algorithm for scheduling parallel applications through DVFS technique, while maintaining the SLA on a cluster platform. The main idea behind EASLA algorithm is to allocate each slack to a maximum set of independent tasks for each task using a compatible task matrix and scale frequencies down to minimize energy consumption within certain extension rate of makespan mutually accepted by user and service provider.

3 System Model

This section describes the target system and application models and introduces the relevant energy models.

3.1 DVFS Model

New Processors equipped the DVFS technique to reduce the power and energy consumption in data center. It allows processors to operate, using a discrete set of voltage and frequency pairs that is (v_j, f_j). The system enables us to deal with the heavy load of tasks with a useful performance by adjusting the voltage and frequency at the same time. Due to specifically needed increase in both the voltage and frequency, a possible lowest rate in each one in terms of saving energy, through Equation (3.1), where k indicates a total number of operating points in a processor, thus:

$$(v_j, f_j) = \{(v_{\min j}, f_{\max j}) = (v_{1j}, f_{1j} < (v_{2j}, f_{2j} < \cdots < (v_{kj}, f_{kj}) = (v_{\max j}, f_{\max j})\} \tag{3.1}$$

where (v_{kj}, f_{kj}) is either the voltage or the frequency of a processor j that exists in a level k.

3.2 Estimating Makespan

Makespan ($C_{\max}$) is defined as the amount of time, from start to end for completing a set of sequences. The most appropriate effort of a scheduling algorithm obtains the minimization of makespan. Equation (3.2) indicates how to obtain makespan.

$$C_{\max} = \left[\max(taski.t^{end}) - \min(taskj.t^{st})\right], \qquad 1 \le i, j \le n \tag{3.2}$$

The critical path in a DAG constitutes the longest path from an entry to exit point of a node in a graph. In the case of dependency of tasks a directed edge of a parent task will pursuit the other task. It then should determine all possible paths. Against the others, the path that includes a longer duration will indicate an appropriate run time of the efficient application. As such, the tasks will indeed stick to the defined execution time. Furthermore, it can be assumed that the tasks of the critical path may assign to the processing resources with high voltage and frequency; on the other hand, the processing elements with lower voltage and frequency may choose to allocate to those that exist in the DAG G but not the critical path. This will decrease consuming the energy/power of computation for the processors of a data center. Accordingly, Equation (3.3) can be expressed as follow. Where makespan can be equal to the Critical Path Length (G).

$$C_{\max}(G) = CPL(G) \tag{3.3}$$

3.3 Energy Model

Scheduling of dependency tasks for computational systems needs consumption of energy that is equal to the total energy consumption of the processors in both tasks execution and data communication to another processor in a communication network.

3.3.1 Energy for Computation

Nowadays, CMOS technology design is commonly used to build processors in which power is consumed in one way, either static or dynamic consumption of energy that is obtained by Equation (3.4):

$$P = P_{dynamic} + P_{static}. \tag{3.4}$$

Static power consumption, i.e. the main source of static current, is leakage current and reverse based PN junction when there is no circuit activity, whereas dynamic power consumption involves charging and discharging of capacitances when inputs are active. P_{static} Can be eliminated from the equation, regarding the energy consumption for the execution of the parallel tasks based on the energy for computation in processors and the energy in communications among processors. Thus, Equation (3.5) is for the calculation of dynamic energy consumption in a processor:

$$P_{dynamic} = ACv^2 f \tag{3.5}$$

where A indicates the percentages of active logic gates with the capacity of dynamic charging, where C denotes a full load of a capacitor, where v is voltage and f is the frequency of the processor.

DVFS based processors meet the maximization of Power consumption ($P_{proc.highest}$) when the operation is carried out with the highest possible voltage ($v_{highest}$) and frequency ($f_{highest}$). Therefore, one can calculate the active power consumption of a processor based on a set of voltage and frequency (v_j, f_j) by Equation (3.6):

$$P_{proc\,j} = P_{proc.highest} \times \frac{v_j^2 \times f_j}{v_{highest}^2 \times f_{highest}} \tag{3.6}$$

$$P_{proc.highest} = ACv^2_{highest} f_{highest}.$$

Since the proposed algorithm adopts the task duplication strategy for scheduling a DAG with n tasks on DVFS-enabled processors, the total energy consumption can be calculated through Equation (3.7).

$$P_{processors.active} = \sum_{i=1}^{n} P_{proc.highest} \left(\sum_{j=1}^{k} \frac{v_j^2 \times f_j}{v^2_{highest} \times f_{highest}} \right) \tag{3.7}$$

$$E_{processors.active} = \sum_{i=1}^{n} P_{proc.highest} \left(\sum_{j=1}^{k} \frac{v_j^2 \times f_j}{v^2_{highest} \times f_{highest}} \right) \times et\big(t_i, p_m(v_j, f_j)\big).$$

Equation (3.8) presents the obtaining of the energy consumption of the processors in idle durations in which m is the number of processors and makespan is the maximum time for completion of tasks by processors, also known as scheduling length.

$$E_{processors.idle} = ACv^2_{lowest} f_{lowest} \Big(|m| \times makespan - \sum_{j=1}^{k} et(t_i, p_m(v_j, f_j) \Big). \tag{3.8}$$

Finally, the total energy consumed by processors to execute the task dependency graph can be obtained through sum of Equation (3.7) and (3.8) as follow:

$$E_{processors} = E_{processors.active} + E_{processors.idle} \tag{3.9}$$

3.3.2 Energy for Communications

Since the processors in each datacenter have been assumed to be homogeneous, the data transfer speed and power consumption are identical (Equation (3.10)).

$$ec_{ij} = PC \times ct(d_{ij}), \tag{3.10}$$

where ec_{ij} indicates the consumed energy for the communications of $d_{ij} \in D$ and where PC expresses the power of interconnect value. Therefore, the total communication energy for the entire network can be calculated through Equation (3.11).

$$E_{Communications} = \sum_{i=1}^{n} \sum_{v_j \in Succ(v_i)} (x_{ij} \times ec_{ij}), \tag{3.11}$$

where x_{ij} is expressed with Equation (3.12) as follow:

$$x_{ij} = \begin{cases} 0 & if\ (t_i^{end}, p_m) = (t_j^{st}, p_m) \\ 1 & O.W \end{cases} \tag{3.12}$$

3.3.3 Total Energy for Datacenter

This can be expressed by Equation (3.13) for a cloud data center can be obtained through sum of Equation (3.9) and Equation (3.11), thus:

$$E_{Total} = E_{dynamic(processors.active)} + E_{dynamic(processors.idle)} + E_{communication}. \tag{3.13}$$

4 The PATCDD algorithm

This section presents an energy aware task scheduling with combination of Clustering and DVFS slack allocation distribution algorithm in cloud datacenters named as PATCDD, which aims reducing the makespan and energy consumptions.

4.1 Power-Aware Task Clustering - Dynamic Voltage/Frequency Scaling Distribution (PATCDD) algorithm

After applying clustering technique on the input DAG, which leads to lower energy consumption and makespan, we intend to further mitigate the energy consumed by processors by determining the critical path and non-critical paths. We also specify the slack time of non-critical tasks, and calculate the voltage and frequency of processors allocated to processing of tasks in non-critical paths as well as idle and communication phases through scaling down DVFS. For this reason, it is essential to first explore the important parameters used in applying DVFS techniques to reduce energy consumption.

4.1.1 Calculation of Slack Time

The parameters are used to calculate the slack time of tasks and determine the critical path. Calculation of Earliest Start Time (EST) is a top-down method, which starts with the first task and ends with the last task, calculated by Equation (4.1).

$$EST(t_i) = \begin{cases} 0 & if\ pred(t_i) = \varphi \\ MAX\Big(EFT(t_j), MAX(EFT(t_k) + d_{ki})\Big)_{t_k \in pred(t_i), d_{ij} \in D} & o.w \end{cases} \tag{4.1}$$

After calculating EST, the Earliest Finish Time (EFT) can be calculated for task t_i by Equation (4.2).

$$EFT(t_i) = EST(t_i) + et(t_i, p_m). \tag{4.2}$$

Moreover, the calculation of Last Finish Time (LFT) is a bottom-up method, which starts with the last task and ends with the first task, calculated by Equation (4.3).

$$LFT(t_i) = \begin{cases} EFT(t_i) or makespan & if succ(t_i) = \varphi \\ Min\Big(LST(t_j), Min(LST(t_k) - d_{ik})\Big)_{t_k \in succ(t_i), d_{ij} \in D} & O.W \end{cases} \tag{4.3}$$

The Latest Start Time (LST) for task t_i is also calculated by Equation (4.4).

$$LST(t_i) = LFT(t_i) - et(t_i, p_m). \tag{4.4}$$

The non-critical tasks in a DAG are distinguished by the presence of slack. Critical tasks have zero slack, while non-critical tasks have slack value this is known as slack time. For each task ith, slack time is calculated through Equation (4.5).

$$\text{Slack time for task} t_i = LST(t_i) - EST(t_i) \ \ or \ \ LFT(t_i) - EFT(t_i). \tag{4.5}$$

5 Experimental results with simulation

This section presents the experimental evaluation of our proposed heuristic algorithm, 'Power-Aware Task Scheduling with Clustering Dynamic Voltage/Frequency Distribution (PATCDD) algorithm. Moreover, we present the comparison of the proposed PATCDD algorithm with the previous works including Power-Aware Task Clustering (PATC) [17], Power-Aware List-based Scheduling (PALS) [17] and Energy Aware Duplication Scheduling (EADUS) & TEBUS [20], in terms of saving energy. Also, we highlight the contrast of PATCDD algorithm with HEFT [14] and RASD [13], underlying the reduction on the execution time or makespan.

5.1 Experimental setup

First of all, we give the settings required for our experiments, including the information of target platform and the power and makespan constraint settings used for the following performance evaluation.

5.1.1 Settings for Constraints of Power and Makespan

To avoid some unreasonable constraint settings that are impossible for an input DAG to meet, we define and specify a lower bound of the power constraint, denoted as Powmin and a lower bound of the makespan or critical path constraint, denoted as CPmin. The maximum amount of reduction in power consumption by processors to execute tasks can be achieved by eliminating all communication overheads. To that end, it is essential to execute all tasks on a single processor. Therefore, the minimum power consumption can be calculated as shown in Equation (5.1).

$$Pow_{\min}(G) = \sum_{\forall C_i \in C} et(C_i). \tag{5.1}$$

The minimum makespan or critical path is represented as CPmin(G) of which the realization depends on the execution of all the tasks on one processor. It causes the makespan constraint due to neglecting the transfer costs. One can obtain this with Equation (5.2) as follow:

$$CP_{\min}(G) = EFT(t_{exit}). \tag{5.2}$$

5.1.2 Randomly generated task graphs

There are many random graph generator tools to generate weighted application DAG, such as STG (standard task graph) [10]. Standard Task Graph Set (STG) is a kind of benchmark for evaluation of multiprocessor scheduling algorithms. STG is proposed for every researcher to evaluate their algorithms under the same conditions covering various task-graph (TG) generation methods including task graphs generated from actual application programs.

In our simulation experiments, Graphs are generated for all combinations of the above parameters with the number of tasks range between 40 and 200, in steps of 40, besides the tasks of 100, 200, 400, and 800 in graphs as the costs of nodes and edges are randomly selected. CCR values are considered as 0.1, 1, 5, and 10, The following Equation (5.3) indicates how CCR is calculated in a DAG.

$$CCR = \frac{\sum_{1 \le i,j \le n} ct(d_{ij}}{\sum_{1 \le i \le n} w_i}. \tag{5.3}$$

5.2 Simulation Results

CloudSim goal is to provide a generalized and extensible simulation framework that enables modeling, simulation, and experimentation of emerging Cloud computing infrastructures and application services, allowing its users to focus on specific system design issues that they want to investigate, without getting concerned about the low level details related to Cloud-based infrastructures and services [5]. We installed CloudSim in an Asus Notebook with Intel core i7-A540UP CPU 2.4 GHz with 8 cores and 4GB memory. We create five datacenters in our simulation, and set 200 virtual machines, each involving three processor. Table 1 shows the details of the three processor types.

Table 1: power consumption for different voltage/frequency of processors [2]

Processors	AMD Opteron 2218	AMD Turion MT-34	Intel Core i3-540
Voltage (V)	1.1, 1.15, 1.15, 1.20, 1.25, 1.30	0.9, 1.0, 1.05, 1.1, 1.15, 1.2	1.125, 1.125, 1.2, 1.2, 1.3, 1.3, 1.375
Frequency (GHz)	1.0, 1.8, 2.0, 2.2, 2.4, 2.6	0.8, 1.0, 1.2, 1.4, 1.6, 1.8	3.07, 3.2, 3.4, 3.6, 3.8, 4.0, 4.2
Highest power (W)	95	25	108
Lowest power (W)	26.16	6.25	53

The first set of experiments compares the power saving of algorithms with respect to various DAG size. We compare the PATCDD algorithm with four other algorithms, named PALS and PATC algorithm, Energy Aware Duplication Scheduling (EADUS) algorithm & TEBUS. PALS and PATC are a critical path based scheduling algorithm, which attempts to shortest schedule length through clustering tasks in critical path and employ DVFS technique as regard to a decrease in the consumption of energy and makespan. EADUS & TEBUS use the tasks duplication strategies for scheduling DAG based parallel tasks in a cluster computing to reduce power consumption. Results show the high impact of the parameters involved with the size of DAG and CCR on significantly saving the power. The DAG highly focus on computations and communications in the condition of CCR=10 and CCR=0.1. PATCDD achieves better

power saving among the others. The reason behind lies in that in our proposed method, we try to distribute and allocate a slack time to the tasks belonging to a maximum independent set of tasks in cluster. Table 2 compares the proposed method with the other power-aware scheduling algorithms, in terms of the capability of each to save the energy more efficiently.

Table 2: Saving of Power between PATCDD and four other Algorithms

Algorithms	Methods	Power Saving
PATC [17]	DVFS & Clustering	39.7
PALS [17]	DVFS & ETF scheduling	44.3
EADUS & TEBUS	Clustering & Duplication	16.8
PATCDD	Clustering & DVFS Slack Distribution	45.9

Figure 5.2 show the second set of experiments compares the makespan of algorithms with respect to different number of tasks of 40 and 200, in step of 40 and CCR (0.1, 1, 5). We compare the first phase of our proposed method (PATCDD) with two other algorithms, Reliability Aware Scheduling (RASD) and Heterogeneous Earliest Finish Time (HEFT).

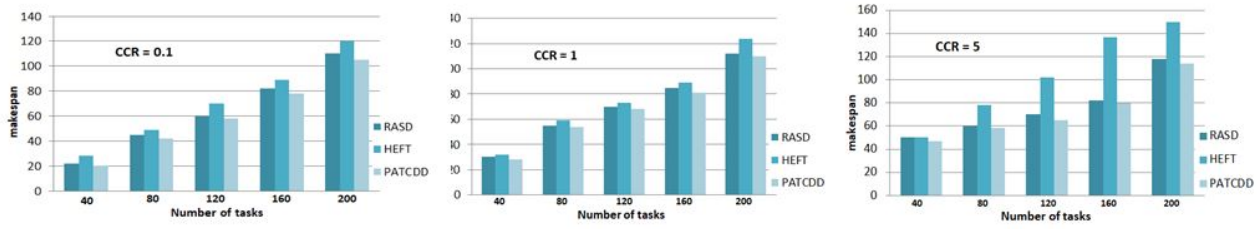

Figure 1: Makespan PATCDD, RASD, and HEFT algorithms for different CCR

6 Conclusion

A power aware scheduling algorithmon DVFS-enabled datacenters can reduce power consumption. In this paper, we proposes power-performance tradeoff scheduling algorithm PATCDD. The proposed algorithms employs the clustering and distribute the slack time for set of non critical tasks in each cluster under a lower voltage and frequency. Eventually, a set of data established for conducting the examinations and also different parameters of the constructed random DAG were assessed to identify the efficiency of our proposed algorithm against with the cluster and duplication based algorithm as well as those with the DVFS technique. The results indicated that the proposed algorithm more significant efficiency of saving power rather PALS, PATC and RASD, HEFT algorithms, respectively, in terms of the power consumption and makespan.

References

[1] B. Barzegar, S. Habibian and M. Fazlollah Nejad, *Heuristic algorithms for task scheduling in cloud computing using combined particle warm optimization and bat algorithms*, J. Adv. Comput. Res. **10** (2019), no. 3, 83–95.

[2] B. Barzegar, H. Motameni and A. Movaghar, *EATSDCD: A green energy-aware scheduling algorithm for parallel task-based application using clustering, duplication and DVFS technique in cloud datacenters*, J. Intell. Fuzzy Syst. **36** (2019), no. 6, 5135–5152.

[3] B. Barzegar and H. Shirgahi, *Advanced reservation and scheduling in grid computing systems by gravitational emulation local search algorithm*, Amer. J. Sci. Res. **18** (2011), 62–70.

[4] A. Beloglazov, J. Abawajy and R. Buyya, *Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing*, Future Gen. Comput. Syst. **28** (2012), no. 5, 755–768.

[5] R. Buyya, R. Ranjan and R.N. Calheiros, *Modeling and simulation of scalable Cloud computing environments and the CloudSim toolkit: Challenges and opportunities*, Int. Conf. High Perform. Comput. Simul. IEEE, 2009.

[6] Y. Ding, X. Qin, L. Liu and T. Wang, *Energy efficient scheduling of virtual machines in cloud with deadline constraint*, Future Gen. Comput. Syst. **50** (2015), 62–74.

[7] S. Fatehi, H. Motameni, B. Barzegar and M. Golsorkhtabaramiri, *Energy aware multi objective algorithm for task scheduling on DVFS-enabled cloud datacenters using fuzzy NSGA-II*, Int. J. Nonlinear Anal. Appl. **12** (2021), no. 2, 2303–2331.

[8] F. Juarez, J. Ejarque and R.M. Badia, *Dynamic energy-aware scheduling for parallel task-based application in cloud computing*, Future Gen. Comput. Syst. **78** (2018), 257–271.

[9] E. Kabir, J. Hu, H. Wang and G. Zhuo, *A novel statistical technique for intrusion detection systems*, Future Gen. Comput. Syst. **79** (2018), 303–318.

[10] H. Kasahara, *Standard task graph set*, 2004." URL http://www. kasahara. elec. waseda. ac. jp/schedule/index. html.

[11] J. Masoudi, B. Barzegar and H. Motameni, *Energy-aware virtual machine allocation in DVFS-enabled cloud data centers*, IEEE Access **10** (2021), 3617–3630.

[12] Z. Peng, B. Barzegar, M. Yarahmadi, H. Motameni and P. Pirouzmand, *Energy-aware scheduling of workflow using a heuristic method on green cloud*, Sci. Prog. **2020** (2020).

[13] X. Tang, K. Li, R. Li and B. Veeravalli, *Reliability-aware scheduling strategy for heterogeneous distributed computing systems*, J. Parall. Distrib. Comput. **70** (2010), no. 9, 941–952.

[14] H. Topcuoglu, S. Hariri and M.Y. Wu, *Performance-effective and low-complexity task scheduling for heterogeneous computing*, IEEE Trans. Parall. Distrib. Syst. **13** (2002), no. 3, 260–274.

[15] J.D. Ullman, *NP-complete scheduling problems*, J. Comput. Syst. Sci. **10** (1975), no. 3, 384–393.

[16] V. Venkatachalam and M. Franz, *Power reduction techniques for microprocessor systems*, ACM Comput. Surveys (CSUR) **37** (2005), no. 3, 195–237.

[17] L. Wang, S.U. Khan, D. Chen, J. Kołodziej, R. Ranjan, C.Z. Xu and A. Zomaya, *Energy-aware parallel task scheduling in a cluster*, Future Gen. Comput. Syst. **29** (2013), no. 7, 1661–1670.

[18] C.M. Wu, R.S. Chang and H.Y. Chan, *A green energy-efficient scheduling algorithm using the DVFS technique for cloud datacenters*, Future Gen. Comput. Syst. **37** (2014), 141–147.

[19] D. Zhu, R. Melhem and B.R. Childers, *Scheduling with dynamic voltage/speed adjustment using slack reclamation in multiprocessor real-time systems*, IEEE Trans. Parall. Distrib. Syst. **14** (2003, no. 7, 686–700.

[20] Z. Zong, A. Manzanares, B. Stinar and X. Qin, *Energy-aware duplication strategies for scheduling precedence-constrained parallel tasks on clusters*, IEEE Int. Conf. Cluster Comput. IEEE, 2006.