A Study of Design and Development of Energy-Aware Scheduling On Single-Core Systems

Vasanth Kumar Kamathamu[1], Dr. Neeraj Sharma[2]

[1]Research Scholar, Department of Computer Science Engineering, Sri Satya Sai University of Technology & Medical Sciences, Sehore, M.P., India.
[2]Research Supervisor Department of Computer Science Engineering, Sri Satya Sai University of Technology &Medical Sciences, Sehore, M.P., India.

Abstract

Over the years, the rapid processing of computational problems in the fields of science, engineering, commercial applications and industrial applications have been shown in parallel computing systems. The careful planning of these applications' concurrent activities is one of the crucial research topics examined during the last several decades. The issue of work scheduling is the processing of parallel application tasks on existing processors to minimise the total length of time. Heterogeneous computing has taken up significant momentum during the past ten years in order to meet the constantly increasing requirements of such systems for performance. The traditional goal of scheduling parallel machine algorithms is to solve the min-max issue, in order to minimise expenses for communication between processes and maximise the use of parallelism in the application. Recently, however, attention has also been given to energy consumption issues in planning algorithms. As the effective use of processors underlies the energy used by the programme may be significantly reduced, energy saving is the newest element in the classic scheduling issue.

Keywords: *Energy-Aware Scheduling, Single-Core Systems, computational problems, commercial applications, machine algorithms, energy consumption*.

1. INTRODUCTION

Energy is a valuable resource of today's modern economy. For energy conservation and sustainability, energy efficiency must be achieved, which may be seen as minimising energy use for a certain amount of labour. This needs methods of achieving higher energy efficiency by correctly arranging tasks across machines and by increasing the energy efficiency of each unit separately. Scheduling theory is an important area of study which has attracted academics' attention for decades; much of the scheduling work has been done with time-related goals. As energy is so scarce and closely linked to cost, in certain cases it may be regarded more essential than time, i.e. it could be preferable to accomplish tasks with little energy instead of doing them fast. Obviously, it is essential to schedule tasks in a manner that the overall energy consumption of the system is minimum or at least modest. In this article, we propose a generic theory for offline energy efficiency planning that does not deal with domain-specific problems or technology.

Our work is used to systems whose machines have varied working and idle power consumption but comparable capabilities. The speeds at which machines may do jobs may vary but are not time-different, i.e., each machine works at the same pace. All machines are linked to one other so that a task may be moved from one machine to another. The number of machines in a system has no fixed limit. Machines and tasks to be carried out are independent and thus any work may be done on any machine in any sequence. Every machine features a set working power (the energy it consumes while loaded) and a fixed idle power (the power it consumes when running idle). Given a number of linked machines and independent tasks, our findings indicate how jobs are to be allocated on the machines to minimise the overall energy usage. Unlike prior work, however, we concentrate on precise outcomes instead of utilising heuristics for scheduling.

We propose a general system model that provides a summary of the essential energy efficiency scheduling features. We next classify the energy-efficient scheduling issues for various systems and analyse their complexity. This enables analyses which, given the energy requirements of the machines, regulate the relative allocation of work among machines to maximise energy efficiency. We develop algorithms for various kinds of systems using these findings. We provide approximation methods along with their limits for problem classes in which scheduling are NP-hard. Intentionally, we do not give specific energy units (typically measured in joules, ergs, kilowatt-hours etc.), power (often watted, etc.), work (usually specified as energy units) or speed (work carried out per unit time); we also do not consider it necessary to be accurate about types of energy or sources of energy. In this way, we stay neutral to certain systems technologies and instantiations and prevent domain-specific differences in our assessments.

2. LITERATURE REVIEW

Jiang Xiaowen, Huang Kai, Zhang Xiaomeng, Yan Rongjie, Wang Ke, Xiaolang and Xiaolang Yan (2014), Recently, energy optimization has been explored for periodic applications on the critical time/safety multiprocessor systems. An important characteristic of system applications is that certain jobs are strictly periodic, while others are not strictly regular, i.e. the start time gap between two consecutive occurrences of the same work is not set until task deadlines may be fulfilled. However, the energy-efficient scheduling of such systems applications has seldom been studied. In this article, we concentrate on the issue of static scheduling multiple periodic applications using strictly and non-strictly periodic safety/time sensitive multiprocessor energy reduction systems. The issue is that strictly as well as non-strictly regular activities have to be tackled intelligently in order to optimise energy usage. We propose a new practical task model to describe the unique characteristic of each work and construct a model-based energy-efficient scheduling issue.

Sergio Nesmachnow, Bernab'e Dorronsoro, Pascal Bouvry and Santiago Iturriaga (2013), This paper presents ME-MLS, an effective local search multi-threading method to solve the multi-target planning issue in heterogeneous system systems. We contemplate minimising both the targets of make-up and energy usage. The suggested technique follows a totally multi-target strategy utilising a Pareto-based domain search conducted with multiple threads simultaneously. The experimental study shows that the novel multiple-threading method performs quick and precise deterministic heuristics based on the conventional MinMin. The novel ME-MLS technique may enhance both the efficiency and energy consumption goals in decreased delivery times for a wide range of test bed cases, while demonstrating almost linear speed-up behaviour when utilising up to 24 threads.

Houssam Eddine Zahaf. Eddine. Energy-efficient planning of parallel real-time activities on multi-core heterogeneous systems (2016), cyber physical systems (CPS), and the Internet of Objects (IoT) generate unparalleled amounts and a diversity of data that must be gathered and stored before processing on the cloud. When the data reaches the cloud for processing, the chance to trigger a response may be late. One way to address this issue is to evaluate the time-sensitive data at the network edge near to the source. Therefore, the cloud only receives the pre-processed results. This paradigm is referred to as *Fog Computing* or *Edge Computing*. Real-time limitations may arise for critical CPS applications employing fog computing models since the findings need be supplied in a predetermined time frame. In addition, processing may be simultaneously applied on several sub-sets of data using the average parallel programming methods in many important applications of CPS.

Rashmi, Dalvi (2014), Wireless Sensor Network (WSN) is an ad hoc class that has the capacity to organise itself, analyse network data and monitor the environment without any attention. Sensor-cloud is a heterogeneous WSN cloud. It is interesting because it can alter the wireless sensor network computation paradigm. In Sensor-Cloud, several WSN owners work together to offer a cloud service to profit from underused WSNs. Sensor Cloud customers may simply rent sensing services to save ownership costs, allowing for the costly use of massive sensor networks. Sensor Cloud provides resource sharing and the scalability of virtual sensors. It summarises many platforms and therefore creates the feeling of a unified network. Users of various apps may also need data based on distinct requirements in a multi-application environment. Hence the scheduling method is needed in WSNs that supports as many users as possible of different applications. We have suggested a scheduling mechanism for Sensor Cloud applications.

Selome Tesfatsion Kostentinos (2012), Energy efficiency in data centres, due to problems connected with energy usage, such as capital costs, operational expenditures and environmental implications, has become an increasingly significant concern. While energy loss due to the under-optimized usage of non-IT equipment has largely been minimised via the use of best-practice technologies, energy waste management in IT equipment still has to be developed and implemented using energy-aware resources management systems. This thesis focuses on developing techniques for allocating resources in data centres, in order to increase energy efficiency. The thesis uses three methods to enhance efficiency in terms of power and performance optimization: the scaling up of virtual machinery (VM) and the processing of servers to decrease energy consumption. In order to accomplish these objectives, the first part of the thesis offers models, algorithms and methods that decrease energy consumption through the use of VM scaling, VM sizing for CPU and memory, CPU Frequency adaptability and server-level hardware power capping.

IEEE Member, Gang Zeng Member, IEEE Membership, Xiongren Xiao Membership, Renfa Li Senior Member, IEEE, Keqin Li and IEEE Membership (2017), Minimizing energy consumption is one of the main design objectives for heterogeneous distributed systems. Using state-of-the-art algorithms, the problem of minimising the energy consumption of a parallel, real-time application with preceding limited tasks on a heterogeneous distributed system is studied by introducing Last Finishing Time (LFT) to reclaim the time of slack by means of an energy efficient design optimization technique (DVFS). However, the use of DVFS alone is inadequate and the decrease in energy consumption is reduced since the frequency is reduced in practise. Moreover, these studies only reduce energy consumption via a local, energy efficient scheduling method, such as decreasing energy consumption on a fixed processor for each job, rather than by using an energy-efficient scheduling approach globally. With the combined non-DVFS and global DVFS enabled energy efficient schedulation algorithms this research addresses the issue of reducing the energy consumption of a real-time parallel application on heterogeneous distributed systems. The Non-DVFS Energy Efficient Scheduling (NDES) method is solved by adding the term slack idea to minimise energy usage and meet the time limits.

Julia Hage, Nizar Hamadeh, Hussein El Ghor, and Rafic Hage Chehade (2017), In the last decades, owing to the enormous advances of integrated circuit technology, we have witnessed a rapid technological scaling. As enormous integration proceeds, the energy consumption of the IC chips rises exponentially, thereby increasing the dependability of the system further. This presents major difficulties in the design of autonomous systems in real time. In this document, we address the issue of developing sophisticated, time-sensitive, energy-efficient and fault resistant design algorithms. To that aim, we first examining the issue of creating real-time uniprocessor scheduling technologies that reduce energy usage while yet tolerating up to k transient failures to maintain the dependability of the system. The first scheduler is an extension of the optimum defective, energy-efficient scheduling method called ES-DVFS. The second method is intended to improve energy savings by setting aside sufficient time to recover from defects.

3. ENERGY-EFFICIENT CLOUD COMPUTING

The evolving Internet technologies have created the cloud computing paradigm, a concept that allows flexible, cost-effective and scalable allocation of resources while making end-users easy to use. The growing demand for cloud services has led to a rapid increase in the number of computer resources and large-scale data centres.

Datacenters may consist of thousands of server units occupying significant space and using enormous electricity. There are 100,000 m2 of data centre space, 12,000 server racks and 1 host servers. The Natural Resource Defense Council estimates that 12,000,000 datacenter servers will provide almost all the US internet activity in 2014. The number of data centres "hyperscale," the biggest data centres, is projected to increase from 338 to 628 by 2011 by the end of 2016. The energy usage of datacenters will also grow dramatically: one estimate estimates that the share of power used by data centres globally would rise from 1.3% in 2010 to 8% by 2010. The consequences of excessive power usage include high cloud energy prices, major investment in plants to cool datacenter computers, and high carbon dioxide emissions. Data centres worldwide suffer $27 billion in yearly energy costs for electricity datacenters. Annual CO2 emissions would rise from 116 million metric tonnes in 2007 to 257 million metric tonnes by 2010, and carbon emissions from datacenters are projected to equal or surpass the aviation sector by 2010.

Energy efficiency for cloud data centres is becoming more essential. Their increasing size and widespread usage have caused considerable energy consumption. It is essential to examine it thoroughly and discover the causes behind it before the issue is resolved. This study presents the ideas of cloud computing and virtualization as its technology. We examine the issue of energy efficiency in cloud data centres further by investigating the main sources of energy waste, offering various methods for energy conservation and introducing Cloud energy monitoring and modelling. Finally, we emphasise the emphasis and direction of this argument.

4. DEVELOPING AN ENERGY-EFFICIENT REAL-TIME SYSTEM

In recent decades, our civilization has witnessed enormous development in embedded technologies. They are constructed effectively, attracting minimal notice to their existence while fulfilling their function yet can scarcely be overlooked in society. In the operation of modern civilization, embedded systems play an important role from electronic calculators to heavy construction equipment. In recent years, its importance has increased and does not indicate a tendency to slow down in the future. An embedded system is usually intended to have particular functions in a larger electrical or mechanical system for solving certain problems. It does not have a display or keyboard, but may be incorporated into system to quickly utilise the user interface, unlike a standard purpose computer. They usually include at least one CPU, both RAM and ROM, as well as certain I/O devices. Real-time applications on embedded systems become more computational with the development of technology.

Powerful and energy-efficient designs play a significant role for contemporary implementations of hardware and software to extend battery life or to reduce maintenance power costs. The sophisticated technology of VLSI circuit designs allows contemporary processors to function at various power voltages using the DVS method and may thus alter the frequency (speed) of the processor. DVS CPUs and power-aware subsystems are currently used in many contemporary computer systems with high energy efficiency needs. The Intel StrongARM SA1100 processor and Intel XScale are well-known processors for example in embedded systems. Intel SpeedStepR and AMD Power NOW!TM technologies offer DVS for laptop computers to extend battery life. Intel Corp. supports several voltage levels for operations on I/O devices. In general, different voltages of supply lead to various processing rates. The slower the speed, the lower the dynamic energy uses.

Task response times are an essential non-functional requirement of the system in many applications (average or worst case). Embedded real-time systems, for example, must perform tasks in order to preserve system stability before their deadlines. For demanding real-time activities on DVS processors, energy efficient scheduling aims to reduce energy usage, while all real-time tasks are carried out in time. We utilised the words task and job in our following discussion, since they are frequently used in real-time literature on systems. A job is an instance of signal processing, computing, or data recovery, etc. A task is a series of tasks with same or comparable features and times. A periodic task is an endless series of task instances, or jobs, where each task takes place on a regular basis.

In the last 10 years, much emphasis has been paid to energy efficient job scheduling with various deadlines. In order to decrease the job's consumption of energy for a given system, the execution route of the task may be optimised to lower the effective switch capacity or to slow down task speeds, e.g. the above optimization could be done off-line during the task compilation process. The latter optimisation, as examined in this article, is carried out by the task planner and the consignor.

Speedy advances in semiconductor technology led to greater chip density and frequency of operation, making today's systems more complicated and power consuming. The microprocessor power consumption has risen approximately linearly with a frequency product throughout the years. Such a high energy consumption necessitates costly packaging and cooling methods, since inadequate cooling leads to a high working temperature that aggravates numerous silicon faults. Manufacturers are now being pressured to minimise the dissipation of power in their goods in order to preserve the dependability of their products and avoid costly packaging and cooling methods.

5. DVFS ALGORITHMS

DVFS-based algorithms depend on the system's ability to change the voltage and frequency of the supply to the CPU (therefore the speed) to decrease the power consumption while fulfilling the real-time limitations. Historically, these methods were the initial attempt to addressing the energy management issue, since dynamic power consumption has been recognised as far more significant than leaky (static) use in CMOS circuits.

Most early DVFS algorithms used a power function equal to P(s) = sα (2 to α to 3), ignorant of leaking power implicitly. The lower the speed, lesser the energy used by such a power function; thus, this power model favours the algorithms with the lowest speed that can always fulfil deadlines, without any idle periods.

The task slack refers to the CPU time not used before its deadline. Therefore, any task ̇i may be calculated offline as slacki = Di – Ri, where Ri is Ti's worst-case reaction time. When running the task early, additional slack (called dynamic slack) may be accessible without using WCET.

The DVFS solutions may also be classed as in-task algorithms. In inter-task algorithms, when a job is dispatched it is guaranteed to perform at the same pace until another (high-priority) work completes it or pre-empts it. If the implementation is resumed (after preemption), the scheduler may change the pace at that moment, considering the slack available. The cross-task algorithms constitute the bulk of the existing DVFS solutions, since they need just information on the WCET of jobs, with a minimal overhead runtime. If, on the other hand, information on the execution time of the task, in particular its likelihood distribution is known, then the pace of the operation may be adjusted at certain times while it is in process. This is the fundamental concept underlying the intra-task algorithms [XXMM04, XMM05, SKL01], where the job begins to run at a low speed level (relying on its early completion being more probable than the worst-case situation), which then progressively increases at established POSs as it continues to run. Therefore a speed schedule is calculated offline for each activity, which shows how fast its tasks are given and at what moment. However, the compiler must also generate code to enable the application to make system calls to the operating system during well established PMPs when performing work, and they involve a greater overhead due to frequent changes in speed. Intra-task algorithms aim at minimising the expected dynamic energy consumption. The remainder of this section presents a survey of the DVFS algorithms most important to scaling speed, separated by the kind of slack they use: static slack, dynamic slack, or both.

1. Static slack reclaiming

Yao et al. [YDS95] is one of the first publications on DVFS energy-aware planning. Three methods were given by considering aperiodic jobs, continuous CPU speed, no overhead speed scaling, minimal power usage during idle periods and task time calculation in inversely proportion to the CPU speed (C(s) = C/s). The first method consists of a recursive time interval identification with the highest computational density (defined as the sum of CPU cycles of the tasks with arrival and deadline within the interval, divided by the length of the interval length). Specifically, the algorithm finds the highest intensity interval, determines the intensity value of the CPU speed for that period, then revisits the calendar for the rest of the execution intervals. The offline method has been shown optimum for n regular tasks and has an O(n log2n) complexity. A second online algorithm examines work that may happen dynamically. The system republished the best schedule at each time the new and pending tasks were taken into consideration. The third method (AVR) sets the speed for each instant equal to the sum of density of tasks whose time and date are considered. While the complexity of AVR is less than the previous optimum methods, time limits may be missed. Indeed, because the speed is equal to the total of the worst-case use of active jobs, the processor may be substantially reduced if few active tasks are involved, such that the remaining work cannot be completed when new work is performed which leads to a missing date.

2. Dynamic slack reclaiming

All algorithms discussed here are EDF-based and assume that the calculation times are linear to the speed (C(s) = C/s). Lee and Shin [LS04] developed OLDVS, an algorithm that accumulates dynamic slack due to early completion and leverages it in order to reduce CPU speeds in order to finish the present job at the same time as the worst-case workload is scheduled. The concept was enhanced in [GSL07] by the intra-task algorithm OLDVS*, which splits each work performance into two parts: the first portion is performed at a low pace and if the speed is not completed by the end of the first half the speed is raised. This method is based on the fact that in the initial portion the likelihood of completing the task is considerably greater than in the second half. Both methods assume discreet speeds, little energy usage in idle periods and 0% overhead switching.

3. Dynamic and static slack reclaiming

All the algorithms reported here consider periodic tasks whose computational times scale linearly with the speed (C(s) = C/s). Moreover, the speed scaling overhead is considered negligible and the power consumption is modelled as $P(s) = \beta \cdot s^3$.

Three algorithms were suggested by Pillai and Shin [PS01], which took into account both EDF and RM planning policies. The first method, called Static Voltage Scaling (SVS), is offline and operates solely with the static slack: when the system begins, the operating speed is set to the lowest possible velocity, which ensures the feasibility of the given job. A cycle-conserving algorithm is then added (cc-EDF and cc-RM). At every scheduling event, the algorithm lowers the operating speed to the lowest level, ensuring time restrictions for finished tasks by the actual execution time and WCET information for future jobs. Notice that if the actual working load is equal to the worst scenario, the cc-EDF algorithm produces a schedule identical to the SVS schedule. The latest suggested algorithm, Look-Ahead RT-DVS (LA-DVS), is exclusively operated by EDF and seeks to reduce the present (earliest) workflow as much as feasible while maintaining other work deadlines. Thus, although the actual speed may be very low by the next deadline, it may be required to do subsequent tasks at high speeds to fulfil their time limitations in case the present work takes (almost) its WCET. However, due to numerous early task completions in practise, this adverse effect is substantially minimised.

6. DPM ALGORITHMS

DPM-based energy management techniques are based on the concept that the CPU is running at low power (sleep) states. One major issue in DPM research is ensuring transitions are positive in terms of energy savings; there is a minimum gap (called breach time) which depreciates the overhead time and energy associated with each transition. In fact, a typical method is to utilise the task delaying process, which delays the performance of ready tasks as much as possible by leveraging the damaged system, compacting busy times and providing lengthy idle intervals. This also reduces the amount of runtime transitions and overhead. On the other hand, maximum care should be made to prevent breaches of the temporal limitations in real time systems while using the recurrence method. The following part presents the most intriguing offline and online DPM methods in the literature.

The break-even timings for the CPU are explicitly considered in all the methods described in this section. Although some studies only examine a single low power state, it may easily be expanded to systems with several low-energy countries by using the "deepest" inactive state with a break-even time less or equal to the available idle time.

1. Offline DPM algorithms

Rowe et al. [RLZR10] introduced two methods to synchronise work periods for the purpose of compiling task executions (i.e., to combine processor idle times whenever possible). The framework assumes a system lacking the DVFS functionality. The first algorithm, Rate-Harmonized Scheduler (RHS), presents the harmonisation idea (TH). When the job arrives, the scheduler is informed only in whole multiples of the harmonisation periods. The harmonised time is calculated according to the shortest time. For example, if the effective arrival time is 3.5 and the harmonisation time is 1, this arrival will only be considered by the scheduler at time 4. Since all arrivals are examined for multiple integers of the harmonising periods, if there is no job to do, the CPU may be in sleep until the next period. The method included priority tasks whose priorities are set forth in the Monotonic Policy Rate. Although it is possible to verify precise schedulability by calculating the worst case response time using the time requirement analysis, the use limit for schedulability is often reduced to 0.5. The 2nd method, termed RHS (ES-RHS), sets a new job with a time-limit equal to TH (highest priority). Their calculation time is assessed by consideration of TH and the replacement use. The new job allows the processor to sleep if it is called in and its computer budget is longer or equivalent to the break-even time. The primary benefit of ES-RHS for RHS is that the idle times produced by task early ends increase the sleep interval throughout the following period. In this manner, many small idle periods are coupled with a single lengthy interval, which gives RHS an advantage. There are two low-power modes, idle and sleep, with a short and a lengthy break-even time. Furthermore, there is an actual implementation on a sensor node.

2. Online DPM algorithms

Lee et al. [LRK03] presented two leakage control methods, both under dynamic (LC-EDF) and fixed (LC-DP) priority programming, for the replacement as long as feasible of the job. Both algorithms take on regular work with deadlines and a system without DVFS. The primary concept behind the algorithms is to calculate the maximum time the work may be delayed without missing the deadline for each job arrival. Whenever the CPU becomes idle under EDF scheduling, LC-EDF calculates the maximum length of time to postpone the task with the earliest time of arriving (Ţk) using the following equation:

$$\sum_{i\in\{1,\ldots,n\}/\{k\}} \frac{C_i}{T_i} + \frac{C_k + \Delta_k}{T_k} = 1.$$

Then, the system is put to the low-power state (procrastinated) for Δk time units. If another higher-priority task τj with absolute deadline shorter than the τk's deadline arrives before the end of the procrastination interval, the procedure is executed again, by considering the length of the idle interval already elapsed, δk, and obtaining the new value of the procrastination interval Δj through the following equation:

$$\sum_{i\in\{1,\ldots,n\}/\{k,j\}} \frac{C_i}{T_i} + \frac{C_k + \delta_k}{T_k} + \frac{C_j + \Delta_j}{T_j} = 1.$$

For fixed-priority systems, the authors use the dual-priority system [DW95] to calculate the duration of the delay period. More specifically, the extra sleep time is calculated as Yi (relative time less the worst case response time) as the minimum promotion time for the lesser queue jobs. The promotional time of each job is calculated as the difference between its relative deadline and the worst response time from the time demand analysis. The major drawback of this method is that it needs hardware for algorithms to be implemented, and sleep and wake actions are managed. Although early job terminations are not included directly in the analyses, the (non-reflecting) nature of the algorithms may indirectly integrate the dynamic slack in time.

7. CONCLUSION

In this study, an efficient energy scheduling method is given for single processor embedded systems in real time. The proposed scheduling method is able to handle irregular and periodic task charts with previous limitations. The static power reduction component sets the minimal voltage for each activity. The online power reduction component alters clock speed dynamically to take advantage of runtime changes in task performance. It also takes advantage of idle periods by powering the CPU down to minimise processor power usage. The method presented is assessed on many common standards. Experimental findings indicate that the method suggested leads in substantial energy savings. In this study, an energy efficient technology is proposed to solve the problem of ensuring end-to-end and local message deadlines in embedded systems disseminated in real time. The methods presented utilise an extensive traffic description function that gives sufficient information on the worst-case traffic behaviour across the network. The node's service rate is lowered to the extent that messages may be delayed or stretched to their worst-case delays. Furthermore, the traffic pattern is evaluated regularly to assess the optimum feasible service rate to take advantage of the temporal changes in calculation requirements. Experimental findings have shown that the methods suggested lead to significant energy savings. With the development of portable devices, low-power design has become an important factor in the design of current applications. Increasing system functionality tends to be achieved by software and thus system-level power saving methods become essential, particularly at operating system level. Extensive researches have been carried out in the literature on low power system design. Nevertheless, the quest for even superior low power systems will continue for many years to come. This dissertation presents new energy-efficient methods for the planning of real-time activities in single and multi-processor embedded systems.

8. REFERENCES

1. Xiaowen Jiang , Kai Huang , Xiaomeng Zhang , Rongjie Yan , Ke Wang , Dongliang Xiong and Xiaolang Yan (2017), "Energy-Efficient Scheduling of Periodic Applications on Safety-Critical Time-Triggered Multiprocessor Systems", Electronics, ; doi:10.3390/electronics7060098.
2. Bernab´e Dorronsoro, Pascal Bouvry and Santiago Iturriaga, Sergio Nesmachnow (2013), "Energy Efficient Scheduling In Heterogeneous Systems With A Parallel Multiobjective Local Search", Computing and Informatics, Volume 32.
3. Houssam Eddine Zahaf. Energy efficient scheduling of parallel real-time tasks on heterogeneous multicore systems (2016). Computer Science [cs]. Sciences et Technologies. English. fftel-01395879, https://hal.archives-ouvertes.fr/tel-01395879.

4. Dalvi, Rashmi (2014), "Energy efficient scheduling and allocation of tasks in sensor cloud". Computer Sciences Commons, Masters Theses. 7324. https://scholarsmine.mst.edu/masters_theses/7324.
5. Selome Kostentinos Tesfatsion (2017), “Energy-efficient Cloud Computing: Autonomic Resource Provisioning for Datacenters”, Department of Computing Science, ISBN 978-91-7601-862-0, ISSN 0348-0542.
6. Guoqi Xie, Member, IEEE, Gang Zeng, Member, IEEE, Xiongren Xiao, Renfa Li, Senior Member, IEEE, and Keqin Li, (2017), “Energy-efficient Scheduling Algorithms for Real-time Parallel Applications on Heterogeneous Distributed Embedded Systems”, Transactions on Parallel and Distributed System, DOI 10.1109/TPDS.2017.2730876.
7. Hussein El Ghor, Julia Hage, Nizar Hamadeh,And Rafic Hage Chehade (2016), “Energy-Efficient Real-Time Scheduling Algorithm for Fault-Tolerant Autonomous Systems”, Scalable Computing 19(4):387-400, DOI:10.12694/scpe.v19i4.1438.