A GENERAL-PURPOSE AND MULTI-LEVEL SCHEDULING APPROACH IN ENERGY EFFICIENT COMPUTING

Mehdi Sheikhalishahi, Manoj Devare
Department of Electronics, Computer and System Sciences, University of Calabria, CS, Rende, Italy

Lucio Grandinetti, Demetrio Lagan
Department of Electronics, Computer and System Sciences, University of Calabria, CS, Rende, Italy

Keywords: Green computing, Cloud computing, Scheduling, Energy, DVFS.

Abstract: Green computing denotes energy efficiency in all components of computing systems i.e. hardware, software, local area and etc. In this work, we explore software part of green computing in computing paradigms in general. Energy efficient computing has to achieve manifold objectives of energy consumption optimization and utilization improvement for computing paradigms that are not pay-per-use such as cluster and grid, and revenue maximization as another additional metric for cloud computing model. We propose a multi-level and general-purpose scheduling approach for energy efficient computing. Some parts of this approach such as consolidation are well defined for IaaS cloud paradigm, however it is not limited to IaaS cloud model. We discuss policies, models, algorithms and cloud pricing strategies in general. In particular, wherever it is applicable we explain our solutions in the context of Haizea. Through experiments, we show big improvement in utilization and energy consumption in a static setting as workloads run with lower frequencies and energy optimization correlates with utilization improvement.

1 INTRODUCTION

Primary use of energy in ICT is in data centers. Efficient power supply design and evaporative cooling rather than air conditioning are two common ways to reduce the energy consumption; in addition, nowadays work on other areas such as resource management and scheduling to optimize energy consumption (Kim et al., 2007) (Dhiman et al., 2009) in computing paradigms in order to be carbon neutral, more environmentally friendly and reducing operational costs is a hot research topic. In this paper, we present a perspective on energy efficient computing to achieve manifold objectives of energy consumption optimization, utilization improvement and revenue maximization for cloud provider in cloud paradigm. This approach is a multi-level and general-purpose scheduling approach which could be applied to all level of resource management stack in any distributed computing paradigm. There are many approaches, mechanisms and algorithms in the literature for energy efficiency, which most of them are special-purpose. The proposed approach is architected in all layers of computing paradigms and systems in order to be general as much as possible. However, it can be extended or specialized for different environments.

We focus our attention on exploring pricing strategies, policies, models and exploiting the latest techniques and technologies in modern processors to design scheduling algorithms to improve resource utilization by using free spaces in scheduler's availability window and optimizing energy consumption.
Throughout this paper we refer to Haizea (Sotomayor, 2010), it is an open source lease (implemented as virtual machines) management system which supports pluggable scheduling policies, various scheduling algorithms, etc. Being open, flexible and modular for further extensions and enhancements, and a general-purpose VM-based scheduler, motivated us to select Haizea as a tool for explaining where our solutions could be applied. In brief, our paper makes the following contributions in the field:

- We propose an abstract approach as a multi-level and general-purpose approach in energy efficient computing by exploring policies, models and algorithms. We highlight that energy efficient

Sheikhalishahi M., Devare M., Grandinetti L. and Lagan D..
A GENERAL-PURPOSE AND MULTI-LEVEL SCHEDULING APPROACH IN ENERGY EFFICIENT COMPUTING.
DOI: 10.5220/0003380000370042
In *Proceedings of the 1st International Conference on Cloud Computing and Services Science* (CLOSER-2011), pages 37-42
ISBN: 978-989-8425-52-2

scheduling is paradigm based; details of a policy, a model or an algorithm is different in different paradigms. We enumerate components of a distributed system scheduler as frontend policies, core scheduler, information service, and backend policies.

- As part of core scheduler, we develop an energy aware algorithm based on operations in DVFS. Through experimentation we demonstrate how using lower level frequencies as operating point of processors, utilization and energy consumption improves whereas total time increases. According to the simulation results, energy consumption optimization correlates with utilization improvement in a static setting.

Next parts of this paper are organized as follows. After reviewing related works in Section 2, Section 3 goes over contemporary technologies for Energy Efficient computing especially in processors (Section 3.1). Next, Section 4 proposes our multi-level and general-purpose approach in Energy Efficient computing. Then, Section 5 discusses experimental results as early lessons on energy aware operations. Finally, Section 5 presents our conclusions and discusses future work.

2 RELATED WORKS

In this decade, many approaches have been proposed to address the problem of Energy Efficient computing with special attention on energy consumption optimization and utilization improvement. The premise in DVFS works is to shorten the idle period as much as possible using time scaling, since it is always beneficial to do so from energy efficiency perspective. In (Hong et al., 1999), there are some assumptions to simplify DVFS-related operations, when the task arrival times, workload and deadlines are known in advance, these techniques target real time systems. Techniques presented in (Azevedo et al., 2002) require either application or compiler support for performing DVFS.

In (Varma et al., 2003) system level DVFS techniques demonstrated. They monitor CPU utilization at regular intervals and then perform dynamic scaling based on their estimate of utilization for the next interval.In contrast to the aforementioned works, the approaches in (Knauerhase et al., 2008) characterize the running tasks and accordingly make the voltage scaling decisions only for those phases of the tasks where it is beneficial. Further, the policies take DVFS decisions based on how beneficial they are from CPU energy savings perspective. Nonetheless, in (Dhiman et al., 2008) it is shown that doing so does not necessarily result in higher system level energy savings. In (Dhiman et al., 2008) instead of using DVFS, simple power management policies presented based on utilizing the low power modes commonly available in modern processors and memories.

In (Kim et al., 2007), authors proposed power-aware scheduling algorithms for bag-of-tasks applications with deadline constraints on DVS-enabled cluster systems. Eucalyptus (Nurmi et al., 2008), Nimbus (Nimbus, 2010) as Virtual Infrastructure Managers do not support scheduling policies to dynamically consolidate or redistribute VMs. Scheduling component of OpenNebula (Sotomayor et al., 2009) (in Haizea mode) and (Dhiman et al., 2009) are able to dynamically schedule the VMs across a cluster based on their CPU, memory and network utilization. Similarly, scheduling algorithms in (Bobroff et al., 2007) provide dynamic consolidation and redistribution of VMs for managing performance and SLA i.e. service level agreements violations. VMware's Distributed resource scheduler (VMWare, 2010) also performs automated load balancing in response to CPU and memory pressure. However, none of these scheduling algorithms take into account the impact of resource contention and policy decisions on energy consumption.

Authors in (Laszewskiy et al., 2009) present an efficient scheduling algorithm to allocate virtual machines in a DVFS-enabled cluster by dynamically scaling the supplied voltages. vGreen (Dhiman et al., 2009) is also a system for energy efficient computing in virtualized environments by linking online workload characterization to dynamic VM scheduling decisions to achieve better performance, energy efficiency and power balance in the system.

3 TECHNOLOGIES FOR ENERGY EFFICIENT COMPUTING

Modern hardware components such as processor, memory, disk and network offer feature sets (Burd and Brodersen, 1995) to support energy aware operations. Exploiting these feature sets in order to be more energy efficient is a very important and challenging task, for example in modeling cost/performance, in designing algorithms, in defining policies, etc. In this section, we review some of these feature sets.

3.1 Technologies Embedded in Processors

Nowadays processors offer two important features for power-saving, cpuidle and cpufreq (Dynamic Voltage and Frequency Scaling). In the cpuidle feature, there are a number of CPU power states ($C-states$) in which they could reduce power when CPU is idle by closing some internal gates. The CPU $C-states$ are $C0, C1, ..., Cn$. $C0$ is the normal working state where CPU will execute instruction and $C1, ..., Cn$ are sleeping state where CPU stops executing instruction and power down some internal components to save power. The cpufreq is another power-saving method especially when CPUs are in load line, allowing quick adjustment to frequency/voltage upon demand in small interval. The key idea behind DVFS techniques is to dynamically scale the supply voltage level of the CPU so as to provide just-enough circuit speed to process the system workload, thereby reducing the energy consumption.

3.2 Electricity Consumption Formulation

In this section, first we formulate electricity consumption and then we will discuss the effect of using two different frequencies on energy consumption.

To formulate energy consumption model, we consider that processors are homogeneous and DVFS-enabled. They have n operating points for each core in a multicore architecture as the following:

$$VF = \{(f_0, v_0), \cdots, (f_n, v_n)\} \tag{1}$$

If a core runs at frequency f_i it consumes v_i voltage, respectively. According to (Kim et al., 2007) the energy consumption for a computation which uses v voltage to run at frequency f, is as the following (quadratically dependent on the supply voltage level):

$$P_{dynamic} \approx v^2.f \tag{2}$$

$$E \approx E_{dynamic} = \sum_t P_{dynamic}.\Delta t = \sum_t v^2.f.\Delta t \tag{3}$$

Thus, if we have J number of jobs, the energy consumption for scheduling them would be:

$$E \approx \sum_{j=1}^{J} v(j)^2.f(j).t(j) \tag{4}$$

In which $v(j)$ and $f(j)$ are the amount of voltage that is used and the core's frequency while running job j, respectively, from V and S sets defined before. Let's have a job which requires t seconds execution time to complete with a $1GHz$ processor. If this job ran with two different operating points $a(v_a, s_a)$ and $b(v_b, s_b)$, its energy consumption in these two scenarios would be like the following:

$$E_a \approx v_a^2.f_a.t_a = v_a^2.f_a.\frac{t}{f_a} = v_a^2.t \tag{5}$$

$$E_b \approx v_b^2.f_b.t_b = v_b^2.f_b.\frac{t}{f_b} = v_b^2.t \tag{6}$$

so that:

$$v_a > v_b \Rightarrow E_a > E_b \tag{7}$$

If $v_a > v_b$ this means that $E_a > E_b$, so energy consumption in operating point a is greater than operating point b while it will be finished earlier $t_a < t_b$.

4 A MULTI-LEVEL AND GENERAL-PURPOSE APPROACH IN ENERGY EFFICIENT COMPUTING

We believe in order to approach conflicting goals of Energy Efficient computing, a multi-level approach which applies to all the levels of workload's path in resource management stack should be devised. Policies, cost/performance models and algorithms within scheduling domain and pricing schemas in cloud computing paradigm are the components which should become or incorporate energy aware placements, operations, techniques, models, etc.

4.1 Policies

Policies have profound impacts on Energy Efficient computing. Policies could be categorized into three types: general-purpose policies (Dhiman et al., 2009), architecture-specific (or infrastructure-specific) policies (Hong et al., 1999) application-specific (or workload-specific) policies (Kim et al., 2007).

General-purpose policies are those that can be applied to most of the computing models. For instance, CPU/cache-intensive workloads should run at high frequencies, since by increasing frequency the performance scales linearly for a CPU/cache-intensive workload.Architecture-specific policies are defined based on the architecture or infrastructure in which computation happens. Also application-specific policies are defined around applications' characteristic.Workload consolidation (Hermenier et al., 2009) policy is a sort of policy at the intersection of the last two mentioned policies. Mixing various types of workloads on top of a physical machine is called consolidation.Furthermore, consolidation-based policies

should be designed in such a way to be an effective consolidation. In fact, effective consolidation is not packing the maximum workload in the smallest number of servers, keeping each resource (cpu, disk, network and memory) on every server at 100% utilization, such an approach may increase the energy used per service unit.

4.2 Algorithms

Algorithms constitute the next part of energy consumption optimization. At this level, we are dealing mainly with energy aware operations over resources like processor, memory, disk and network. For instance, for a processor we have lightweight operations i.e. decrease/increase freq/volt, moving to an idle-state or moving to a performance-state, and heavy-weight operations such as suspend/resume/migrate and start/stop on virtual machines or turn on/turn off on physical machines.

Algorithms based on cost/performance models are part of the scheduling to model cost vs. performance of system states. A formal cost model quantifies the cost of a system state in terms of power and performance. These models are exploited by the scheduling algorithms to select the best state of a processor, memory, disk and network. Sleep states' power rate and their latency i.e. the time needed to change to and from the running state, are examples of parameters in modeling cost vs. performance. In addition, models should specify how much energy will be saved in state transitions and how long it takes for state transitions. Models are also architecture and infrastructure dependent e.g. internal of Multicore and NUMA systems have different features and characteristics to be considered.

4.3 Pricing Strategy

At the highest level in the cloud interface, we have pricing strategies such as Spot Pricing in Amazon (Amazon, 2010) and recent pricing approaches in Haizea or perhaps game theory mechanisms. These mechanisms apply cloud policies that are revenue maximization or improving utilization. More or less these policies have the same goals, and also they are energy efficient, since they keep cloud resources busy by offering various prices to attract more cloud consumers. A dynamic pricing strategy like offering cheaper prices for applications that will lead to less energy consumption (or higher performance) based on the current cloud status (workloads and resources) compared to the others, is an Energy Efficient pricing schema.

4.4 Autonomic Scheduling

We enumerate components of a distributed system scheduler as frontend policies, core scheduler, information service, and backend policies. In summarizing our approach, as a reference architecture, some components of a distributed system scheduler are enumerated as the following:

- Frontend policies: Admission control and pricing are placed in this component. Job requests pass via these policies before queueing.
- Core scheduler: Queue mechanism, various scheduling algorithms and specific energy aware algorithms such as those dealing with energy aware operations (next Section) constitute core of a scheduler.
- Information service: Scheduler's slot table, availability window, etc. provide various information to be exploited in different components.
- Backend policies: Host selection, mapping and preemption constitute backend policies.

An autonomic scheduling approach could exploit this reference architecture to make various decisions in different components, for example in queue mechansim based on job's characteristic, information provided by information service, and other jobs in the queue, a grouping or affinity mechanism could be implemented to reduce resource contention among jobs; we are studying this research in another work. Nonetheless, more details of this approach is out of the scope of this paper.

5 EARLY LESSONS ON ENERGY AWARE OPERATIONS

In this section, we review two main energy aware operations from the scheduling point of view at the processor level in a detailed manner compared to the previous section.We have added DVFS feature in the Haizea' resource model to support a set of different frequencies and voltages. We also extended duration class of Haizea to keep track of running leases with different frequencies and update the remaining time of a lease accordingly. This section highlights the importance of classifying compute intensive workloads according to their demands and running them on the most appropriate processor (i.e. operating on the frequency required by the classified workloads).

5.1 Experimental Results

In the first experiment, we run Haizea in simulation mode to process 30 days of lease requests from the SDSC Blue Horizon cluster job submission trace (Feitelson, 2010).We have done two separate experiments with two different processors as the processing unit of the nodes, with the following DVFS specifications:

Table 1: Operating Points for processor1.

Perf. state	Frequency	Voltage
P0	3600	1.4
P1	3400	1.35
P2	3200	1.3
P3	3000	1.25
P4	2800	1.2

Table 2: Operating Points for processor2.

Perf. state	Frequency	Voltage
P0	1600	1.484
P1	1400	1.420
P2	1200	1.276
P3	1000	1.164
P4	800	1.036
P5	600	0.956

In each run, we measured the whole experimentation time, sum of all leases slowdown (all-leases-slowdown) and energy consumption according to equation (3) metrics as well as resource utilization.

Tables 3, 4 show these metrics for processor1 and proessor2, respectively.

Table 3: SDSC Blue Metrics for processor1.

Freq	Time(sec.)	Slowdown(sec.)	EC(Joule)	Util
3600	2668402	9870	160560938160	0.68
3400	2690148	12552	149281951131	0.71
3200	2720196	14349	138431042048	0.75
3000	2748824	14473	127989300000	0.79
2800	2829515	20649	117954039168	0.82

This experiment reveals that as frequencies decrease, the running time increases slightly; resource utilization increases and this is true also for slowdown metric. On the other hand, there is a big decrease in energy consumption. In conclusion, if we compare the highest frequency run with the lowest frequency run, we observe that while the running time increases, but there is a big improvement for energy consumption metric; in addition resource utilization increases around 14% which for utilization is a very good gain. Increasing utilization also is an energy efficient approach, since resources would be busy and the waste of energy would become less. Nonetheless, in case of high load it would be better if we run applications with the highest frequency, since during peak times utilization is always high and by this technique computing system could service more jobs. In fact, there is a trade-off between load, utilization, cost, performance, and sustainability. In all, in case of low load which resource utilization is low, taking advantage of running applications with low frequency is a promising technique to fill out availability window of scheduler and increase utilization; on the other hand, in high load times, we could run applications with the highest performance of computing system. Trade-off is a key in all these aspects. Furthermore, we have the same observation for experimentation with processor2.

Table 4: SDSC Blue Metrics for processor2.

Freq	Time	Slowdown	EC	Util
1600	2668402	9870	80180564496.3	0.68
1400	2739271	15832	73407588444.4	0.76
1200	2851453	22414	59276240343.1	0.85
1000	3269150	51268	49327094388.4	0.89
800	4017467	66126	39076964327.3	0.90
600	5336254	85235	33274070266.6	0.91

Table 5: SDSC DataStar Metrics for processor1.

Freq	Time	Slowdown	EC	Util
3600	2654637	9017	399685094928	0.58
3400	2658448	10538	371630049746	0.61
3200	2662737	12051	344615195392	0.65
3000	2667597	16264	318617648438	0.69
2800	2673151	20650	293637262464	0.74

In the second experiment, we studied the same metrics on SDSC DataStar trace. Table 5 shows the aforementioned metrics for processor1.
Interestingly, experimental results are promising with big improvement in utilization and energy consumption as workloads are running with low frequencies.

6 CONCLUSIONS AND FUTURE WORKS

We have proposed a multi-level and general-purpose approach for energy efficient computing. In particular, we have added support for DVFS in Haizea's resource model to do some simulation experiments regarding running workloads with different frequencies in a static setting. Through experiments, we have shown big improvement in utilization and energy consumption as workloads are running with low frequencies and the coincidence of Energy consumption

and utilization improvement.

We discussed different policies. Policies are rules that we define based on the real facts and they optimize related metrics based on our needs. Since OpenNebula as a Virtual Infrastructure Manager (Sotomayor et al., 2009) is integrated with Haizea as an advanced scheduling backend, and they are complementary to each other, we plan to implement different policies in OpenNebula and Haizea.

We believe energy efficient scheduling have to be paradigms specific. Currently, we are exploring consolidation based policies in HPC and cloud paradigms in another work.

ACKNOWLEDGEMENTS

This work was supported by Ministry of University and Research(MIUR) of Italy. We are grateful to Borja Sotomayor for his support and help, development of Haizea and his important contribution in the field. Also, we gratefully acknowledge Ignacio Martin Llorente and Wolfgang Gentzsch for their insightful comments.

REFERENCES

Amazon (2010). Amazon ec2 spot instances. http://aws.amazon.com/ec2/spot-instances/.

Azevedo, A., Issenin, I., Cornea, R., Gupta, R., Dutt, N., Veidenbaum, A., and Nicolau, A. (2002). Profile-based dynamic voltage scheduling using program checkpoints. In *DATE '02*, page 168.

Bobroff, N., Kochut, A., and Beaty, K. (2007). Dynamic placement of virtual machines for managing sla violations. In *International Symposium on Integrated Network Management '07*.

Burd, T. D. and Brodersen, R. W. (1995). Energy efficient cmos microprocessor design. In *Proceedings of the 28th Annual Hawaii International Conference on System Sciences*.

Dhiman, G., Marchetti, G., and Rosing, T. (2009). vgreen: A system for energy efficient computing in virtualized environments. In *the 14th IEEE/ACM International Symposium on Low Power Electronics and Design*. ISLPED '09.

Dhiman, G., Pusukuri, K. K., and Rosing, T. S. (2008). Analysis of dynamic voltage scaling for system level energy management. In *HotPower'08*. the 2008 Workshop on Power Aware Computing and Systems.

Feitelson, D. (2010). Parallel workloads archive. http://www.cs.huji.ac.il/labs/parallel/workload/.

Hermenier, F. et al. (2009). Entropy: a consolidation manager for clusters. In *VEE'09*.

Hong, I., Kirovski, D., Qu, G., Potkonjak, M., and Srivastava, M. B. (1999). Power optimization of variable-voltage core-based systems. *IEEE Trans. Computer-Aided Design*, 18(12):1702–1714.

Kim, K. H., Buyya, R., and Kim, J. (2007). Power aware scheduling of bag-of-tasks applications with deadline constraints on dvs-enabled clusters. In *CCGRID*, pages 541–548.

Knauerhase, R., Brett, P., Hohlt, B., Li, T., and Hahn, S. (2008). Using os observations to improve performance in multicore systems. In *IEEE Micro'08*.

Laszewskiy, G. v., Wangz, L., Youngez, A. J., and Hez, X. (2009). Power-aware scheduling of virtual machines in dvfs-enabled clusters. In *Cluster 2009*. Cluster 2009.

Nimbus (2010). Nimbus toolkit project. http://nimbusproject.org/.

Nurmi, D., Wolski, R., Grzegorczyk, C., Obertelli, G., Soman, S., Youseff, L., and Zagorodnov, D. (2008). The eucalyptus open-source cloud-computing system. In *Cloud Computing and Its Applications'08*.

Sotomayor, B. (2010). Haizea. http://haizea.cs.uchicago.edu/.

Sotomayor, B., Montero, R., Llorente, I., and Foster, I. (2009). Virtual infrastructure management in private and hybrid clouds. *IEEE Internet Computing*, 13(5):14–22.

Varma, A., Ganesh, B., Sen, M., Choudhury, S. R., Srinivasan, L., and Bruce, J. (2003). A control-theoretic approach to dynamic voltage scheduling. In *CASES*, pages 255–266.

VMWare (2010). Vmware dynamic resource scheduler. http://www.vmware.com/files/pdf/drs_datasheet.pdf.