Energy Efficiency of Knowledge-Free Scheduling in Peer-to-Peer Desktop Grids

Aritz Barrondo, Andrei Tchernykh
Computer Sciences Department, CICESE Research Center, Ensenada, Baja California, Mexico
{barrondo, chernykh}@cicese.mx

Elisa Schaeffer
Universidad Autónoma de Nuevo Leon
Nuevo Leon, Mexico
elisa.schaeffer@fime.me

Johnatan Pecero
Faculty of Science, Technology and Communication
University of Luxembourg, Luxembourg
johnatan.percero@uni.lu

Abstract — We address knowledge-free Bag-of-Tasks non-preemptive scheduling problem on heterogeneous grids, where scheduling decisions are free from information of resources and application characteristics. We consider a scheduling with task replications to overcome possible random bad resource allocation and ensure good performance. We analyze energy consumption of job allocation strategies based on variations of the replication threshold. In order to provide QoS and minimize energy consumption, we perform a joint analysis of two metrics. A case study is given and corresponding results indicate that proposed strategies reduce energy consumption without significant degradation in performance.

Keywords: Energy-Aware Scheduling; Knowledge-Free Scheduling; Desktop Grid; Green Peer-to-Peer Computing

1. Introduction

The increasing popularity and diversity of emerging distributed large-scale computing environment systems such as Peer-to-Peer (P2P) [21], Grids, and Clouds implies that their resource management must be able to adapt to changes in their state and requirements to meet desired Quality of Service (QoS) constraints. As systems scale and energy consumption increases, such new technologies have the power to damage our ecosystems. With the growth of computer components density, the cooling problem becomes also crucial, as more heat has to be dissipated per square meter. Another rising concern is the environmental impact in terms of carbon dioxide emissions caused by high energy consumption. Therefore, the reduction of power and energy consumption has become a first-order objective in the design of modern computing systems. Moreover, energy consumption has become one of the main problems that the computer industry has faced due to increasing investment for maintaining the computers [1] and the reduction in performance due to increasing temperatures [2]. Further, a recent study estimates global data center energy consumption could increase 19% this year [15]. Globally, the study identifies 31 GW of data center power consumption.

Energy consumption is not only determined by hardware efficiency, but it is also dependent on the resource management system deployed on the infrastructure.

The scheduling of jobs on multiprocessors is usually focused on the optimization of common performance criteria like total completion time, turnaround time, etc. It is generally well understood and has been studied for decades. Many research results exist for different variations of this scheduling problem; some of them provide theoretical insights while others give hints for the implementation of real systems. However, the power-aware multiple-single-core-machine scheduling problem has rarely been addressed so far. Unfortunately, it may result in bad power utilization.

There are several research efforts on power-aware resource allocation to optimize energy consumption at a single resource computer, a single cluster, or data center. The power usage reduction is achieved by two policies [3]:

(1) Switching off parts of the computer system that are not utilized (dynamic component deactivation) [4, 22];

(2) Dynamic Voltage Scaling (DVS) to slow down the speed of CPU processing [2, 5].

These policies are designed to reduce the energy consumption of one resource, and not consider resources distributed geographically. Usually, they are integrated into resource management systems and scheduling algorithms [16].

We explore benefits of the first method for P2P desktop grids. This policy turns off/on machines so that only the minimum number of machines required to support the QoS for a given workload are kept active [6]. It can be implemented in standard high-performance processors without dynamic voltage scaling. However, some hardware support, such as a Wake-On-LAN network interface, is needed to signal a machine to transition from inactive to active state. This method can be implemented as a separate support server or local server software service to determine whether machines must be turned off or on. The scheduling mechanism must be aware of the state of the machines, so that it does not direct request to inactive nodes without turning them on first.

Only few works consider the energy optimization in P2P grid computing, but none of them consider knowledge- free scheduling. Sharma and Aggarwal [2] used a statistical approach to predict workload energy consumption and used DVS to reduce it. Ponciano and Brasileiro [14] compared idle state, standby and hibernation effects on QoS and energy consumption.

In this paper, we discuss knowledge free scheduling algorithms with QoS based on mean approximation factor.

Hence, we study how our algorithms guarantee to generate a schedule with completion time being within a certain factor of the optimal solution. We analyze energy consumption of different grids considering variety of workloads to find a compromise between better QoS and energy consumption. We also evaluate scheduling algorithms to study relationships between replication, performance, and energy consumption.

We continue this paper by presenting P2P desktop grid scheduling in Section 2. We introduce scheduling model and discuss scheduling algorithms in Section 3, where we also formally present related energy model. We discuss related work in Section 4. We present evaluation method for multi criteria analysis in Section 5. Experimental setups are presented in Section 6, while experimental results are analysed in Section 7. Finally, we conclude with a summary and an outlook in Section 8.

2. Peer-to-Peer Desktop Grid Scheduling

There exist various types of desktop grids depending on their platform (middleware or web based), their scale (Internet or local area network), architecture (centralized or distributed), and their service provider (enterprise or volunteer) [7]. The distributed, Internet, middleware based, and volunteer grids are also known as Peer-to-Peer Desktop Grids (P2P-DG). The main goal of such systems is to provide fast, scalable, and secure service [8]. P2P-DGs are very successful in bringing large numbers of donated computing systems together to form vast resource pools. However, achieving this goal is challenging because current Internet connectivity is far from ideal (due to firewalls and private addressing). Moreover new vulnerabilities appear on a daily basis.

Low cost desktop grids are being developed to help with data processing from scientific projects, for instance, OurGrid [8], and ShareGrid [13]. These types of systems are suited to perform highly parallel computations such as Bag-of-Tasks (BoT) applications that do not require any interaction between network participants.

BoT applications simplify the system requirements since a task completion does not affect other tasks completion, meaning that the grid can deliver execution of applications without demanding any QoS guarantees from the resources. In spite of their simplicity, BoT applications are used in a variety of scenarios, including data mining, massive searches (such as key breaking), parameter sweeps, simulations, fractal calculations, computational biology, and computer imaging [8].

BoTs also simplifies the scheduling model by assuming a single core machine grid. BoT applications are being scheduled with knowledge-free algorithms, such as Work Queue with Replication (*WQR*) in real P2P-DGs [8, 13]. *WQR* uses no information about tasks nor machines and its performance is shown to be not significantly worse comparing with traditional knowledge-based schedulers [9]. However, it consumes more computational cycles, and, hence, volunteer computing resource donors are wasting energy. This kind of waste has a negative effect over the resource provider economy as well as the natural environment [1]. Moreover, taking into account the feasibility of offering cloud computing services that run over this kind of grids, and transforming computer resource donors into computer resource commercial providers [10], efficient energy management would reduce resource providing costs and increment the provider income.

In this paper, we incorporate energy model to the *WQR* scheme, and perform a joint analysis of two metrics: approximation factor and energy consumption. That is, together with energy, we consider minimization of largest completion time of any job in the system to propose strategies that are able to reduce energy consumption without significant degradation in performance.

3. Scheduling Model

We use a model that focuses on some key aspects of P2P-DG. It consists of a set G of m heterogeneous machines N_1, N_2, … , N_m. The number of single core machines can be changed over time. Let s_i be the processing speed of machine i. Let $S = \sum_{i=1}^{m} s_i$ be the total speed of machines or total maximum grid processing power, and s_{max} be the maximum processing speed. The processing speed of machines is unknown.

Addressing an offline scheduling problem: n tasks T_1, T_2, …,T_n of a BoT must be scheduled on m uniform machines. Each task T_j is described by its execution time p_j on the slowest machine that is unknown until the task has completed its execution (non-clairvoyant case). The total execution time of the BoT is denoted by $p = \sum_{j=1}^{n} p_j$, which is fixed for all experiments. The release date of tasks is zero.

A task can be allocated to any machine. We consider that a task is assigned to specific machine only when it is actually available, no local machine waiting queues are considered.

The start time and a completion time of a task j executed in machine i are denoted by s_j , and $c_j = s_j + \frac{p_j}{s_i}$ respectively. The completion time of a BoT is denoted by $C_{max} = max_{j=1,\dots,n}(c_j)$.

The approximation factor of the strategy is defined as $\rho = \frac{C_{\max}}{C^*_{\max}}$, where $C^*_{\max}$ is the optimal makespan.

3.1. Scheduling strategies

We apply *WQR* scheme proposed in [9]. We evaluated four knowledge free scheduling strategies: Work Queue (*WQ*), Work Queue with one Replica per task (*WQR1*), Work Queue with two Replicas per task (*WQR2*), Work Queue with three Replicas per task (*WQR3*).

WQ is a basic scheme. The scheduler randomly allocates a task to an available machine until the BoT is empty. *WQRx* is based on *WQ* by adding x extra replicas for each task. The total number of tasks to be scheduled is $n(1 + x)$. When there are no more tasks to allocate, a randomly chosen running task is replicated and allocated to a machine. When a machine finishes a task (or replica), all replicas of the finished task are canceled. Replication is the key of reducing possible negative effect of bad job allocation on the total BoT completion.

3.2. Energy model

The power consumption $P_i^{core}(t)$ of a core i at time t consists of a constant part $P^{idleCore}$ (power consumed in idle state) and a variable part $P^{workCore}$:

$$P_i^{core}(t) = v_i(t) \cdot (P^{idleCore} + w_i(t) \cdot P^{workCore}), \quad (1)$$

where $v_i = 1$ if the core is on, otherwise $v_i = 0$ at time t, respectively, and if the core is in operational state at time t $w_i(t) = 1$ else $w_i(t) = 0$.

When a core is off, it consumes no power; when it is on, it consumes $P^{idleCore}$ power, even if it is doing nothing. Therefore, the model assumes that power consumption of all system components is essentially constant regardless of the machine activity. Hence, $P^{idleCore}$ includes the power consumption of the core and the power consumption of all other components, including a cooling system. In addition, the core consumes a power $P^{workCore}$ when the core is loaded (in operational mode).

The power consumption $P_i^{machine}(t)$ of a machine i at time t consists of a constant part $P^{idleMachine}$ (power consumed in idle state) and a variable part P_i^{core}:

$$P_i^{machine}(t) = o_i(t) \cdot (P^{idleMachine} + P_i^{core}(t)) \quad (2)$$

$$P^{opGrid}(t) = \sum_{i=1}^{m} P_i^{machine}(t) \quad (3)$$

Where:

$$o_i(t) = \begin{cases} 1 \text{ if machine } i \text{ is on at time } t \\ 0 \text{ else} \end{cases} \quad (4)$$

Hence

$$P^{opGrid}(t) = \sum_{i=1}^{m} o_i(t) \Big(P^{idleMachine} + v_i(t)(P^{idleCore} + w_i(t)P^{workCore})\Big) \quad (5)$$

We consider a startup duration (latency to start up), corresponding with power consumption while turning on and off each component. Therefore, to be effective, a transition has to be done only if the idle period is long enough to compensate the transition overhead. In real systems, there is a limited or no knowledge of the future workload.

For each of the $n_i^{startMachine}$ a machine is turned on, it is ready for operational mode after $T_i^{startMachine}$ time intervals consuming $P_i^{startMachine}$ power.

The energy consumed by the grid is the sum of the energy it consumes for starting up ($E^{startGrid}$) its components and the energy it consumes while its operational state (E^{opGrid}).

$$E^{grid} = E^{opGrid} + E^{startGrid}, \quad (6)$$

where:

$$E^{opGrid} = \sum_{t=1}^{C_{max}} P^{opGrid}(t) \quad (7)$$

$$E^{startGrid} = \sum_{i=1}^{m} E_i^{startMachine} \quad (8)$$

$$E_i^{startMachine} = \sum_{i=1}^{m} \Big(n_i^{startMachine} * T_i^{startMachine} * P_i^{startMachine} + n_i^{startcore} * T_i^{startcore} * P_i^{startcore}\Big) \quad (9)$$

We follow a straightforward approach introduced in [4]: after completion of a job, a core, and machine will be turned off after time interval $d_i^{machine}$ (latency to turn off), if during this interval any other job is not allocated to the machine. The reason for introducing this delay is to avoid frequent switches between on and off states. If within a reasonably short period of time a job arrives, it makes sense to leave the machine on. In this scenario, the grid is on for the duration of the BoT application.

4. Related Work

While power aware scheduling in a single computer is widely studied and shown to be effective, green aspects of Grid computing are beginning to emerge.

Develder et al. [4] study dynamic component deactivation in the EGEE/LCG Grid. Lammie et al. [18] explore energy and performance trade-offs in the scheduling of grid workloads on large clusters. The authors analyze the effect of automated node scaling, CPU frequency scaling, and job assignment. DaCosta et al. [19] present a framework based on three components: an on/off model based on an energy-aware resource infrastructure, a resource management system adapted for energy efficiency, and a trust delegation component to assume network presence of sleeping nodes. Ponciano and Brasileiro [15] investigate energy-aware scheduling, sleeping and wake-up strategies in opportunistic grids. Sleeping strategies are employed to reduce the energy consumption of the Grid during idleness periods; wake-up strategies are employed to choose a set of resources to fulfill a workload demand; scheduling strategies are employed to decide which tasks to schedule to the available machines.

5. Evaluation Method

Since the problem is multi-objective in its general formulation, two criteria are considered: an approximation factor ρ that represents QoS; and mean energy consumption E^{grid}.

A good scheduling algorithm should schedule tasks to achieve high grid performance, while minimizing energy consumption. The problem can be simplified to a single objective problem through different methods of objective combining. There are various ways to model preferences, for instance, they can be given explicitly to specify an importance of every criterion or a relative importance between criteria. This can be done by a definition of criteria weights or criteria ranking by their importance.

In order to provide effective guidance in choosing the best strategy, we perform a joint analysis of two metrics according to methodology proposed in [11], and applied for the grid scheduling problem in [12]. They introduced an approach to multi-criteria analysis assuming equal importance of each metric. The goal is to find a robust and well performing strategy under all test cases, with the expectation that it will also perform well under other conditions, e.g., with different grid configurations and workloads.

The analysis is conducted as follows. First, we evaluate the degradation in performance (relative error) of each strategy under each metric. This is done relative to the best performing strategy for the metric, as follows: $\left(\frac{strategy\ metric}{best\ found\ metric}-1\right)*100$. Then, we average these values, and rank the strategies.

The best strategy, with the lowest average performance degradation, has rank 1. Note that we try to identify strategies which perform reliably well in different scenarios; that is, we try to find a compromise that considers all of our test cases. For example, the rank of the strategy in the average performance degradation could not be the same for any of the metrics individually or for any of the grid scenarios individually.

We present the metric degradation averages to evaluate performance of the strategies, and show if some strategies tend to dominate results. The degradation approach provides the percent improvement, but does not show negative effects of allowing a small portion of the problems to dominate the conclusions.

To analyze possible negative effects of allowing a small portion of the problem instances with large deviation to dominate the conclusions that based on averages, and to help with the interpretation of the data generated by the benchmarking process, we presented performance profiles of our strategies.

The performance profile $\rho(\tau)$ is a non-decreasing, piecewise constant function that presents the probability that a performance ratio $r=\frac{stretegy\ metric}{best\ found\ metric}$ is within a factor τ of the best ratio [17, 23].

The performance profile of each metric is calculated by (10), then they are averaged by (11).

$$\rho_{metric}(\tau)={}^{1}/_{n_{exp}}\ size\ \{results{:}\ result\le\tau\} \tag{10}$$

$$\bar{\rho}_{metrics}=\frac{\sum_{all\ metrics}\rho_{metric}(\tau)}{number\ of\ metrics} \tag{11}$$

Where n_{exp} is the number of experiments and τ varies from the best metric found to the worst metric found. The best strategy should get a higher probability of obtaining a low degradation.

6. Experimental Setup

All experiments are performed using the Grid scheduling simulator *tGSF* (Teikoku Grid Scheduling Framework). *tGSF* is a standard trace based simulator that is used to study Grid resource management problems. We have extended Teikoku to include P2P-DG, energy and replication capabilities. Design details of the simulator are described in [20].

Two fundamental issues have to be addressed for performance evaluation. On one hand, representative workloads are needed to produce reliable results. On the other hand, a good testing environment should be set up to obtain reproducible and comparable results.

We consider five Grid scenarios for evaluation. We fix $S=1000$ to all grids, getting speed heterogeneity of all machines from the uniform distribution $s_i=U\left(10-\left(\frac{hm}{2}\right),10+\left(\frac{hm}{2}\right)\right)$. For 5 grids, we set *hm* equals to *1*, *2*, *4*, *8*, and *16* [9]. Machine energy consumption is presented in Table 1.

Table 1. Energy Consumption and Time Interval for Operational Modes.

Notation	Value
$P^{workCore}$	19.53W
$P^{idleCore}$	8.26W
$P^{idleMachine}$	175W
$P_i^{startMachine}$	201W
$T^{startMachine}$	89s
$d_i^{machine}$	10s

We consider 20 types of workloads setting all BoT sizes to $p=3600000$. 20 types of applications are divided in four groups that are characterized by different mean execution time of tasks ($\bar{p}_j$) in each BoT. $\bar{p}_j$ is set to 1000, 5000, 25000 and 125000. The variation of p_j of 0%, 25%, 50%, 75% and 100% relative to $\bar{p}_j$ of the group with uniform distribution is considered for the five subgroups in each group. We perform an analysis of scheduling strategies under Grid resources heterogeneity, tasks heterogeneity, and the granularity of the BoT. Varying $\bar{p}_j$, the number of tasks in BoT is changing (Table 2). Hence, we evaluate different Grid scenarios varying workload from 36 tasks per machine to 0.29 tasks per machine.

Figure 1 shows some details of the used workload used in our study: minimum, maximum, quartile 25%, quartile 75%, and median of the tasks group execution time. Figure A in Appendix shows a histogram of the task group size distribution in the workload. We can see that the task resource consumption demand is not equally distributed in the BoTs in order to simulate different scenarios.

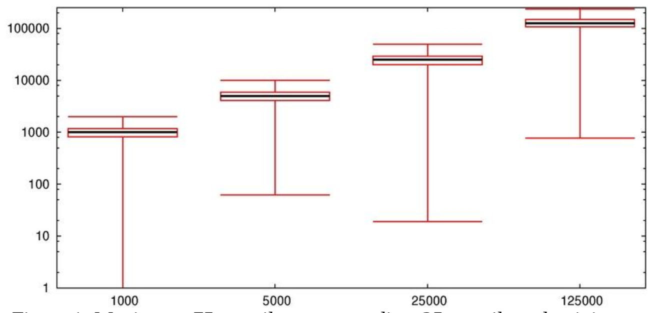


Figure 1. Maximum, 75 quartile, mean, median, 25 quartile and minimum of BoT with $\bar{p}_j = 1000, \bar{p}_j = 5000, \bar{p}_j = 25000, \bar{p}_j = 125000$

Note that, by fixing S and p, difference in completion time can be credited exclusively to the scheduler. Further, in our evaluation, we use the lower bound of the optimal makespan $\breve{C}^*_{max}$ instead of the optimal makespan C^*_{max} with $C^*_{max} \geq \breve{C}^*_{max} = max\left\{\frac{max_j(p_j)}{s_{max}}, \frac{\sum_{j=1}^{n} p_j}{S}\right\}$ as we are, in general, not able to determine the optimal makespan.

Table 2. Workload

$\bar{p}_j$	n	Mean number of tasks per machine
1000	3600	36
5000	720	7.2
25000	144	1.44
125000	29	0.29

7. Experimental Results

This section presents an analysis of the simulation results of BoT scheduling strategies using performance metrics described in Section 3, experimental setup and workload described in Section 6, and evaluation method described in Section 5.

The experimental evaluation of the strategies is presented in two steps. In the first step, we compare strategies for each metric separately, and present their degradations in performance. Then, we perform a joint analysis of these metrics and present their degradations and ranking considering all metrics average. The best strategy, with the lowest average performance degradation, has rank 1. Note that we try to identify strategies which perform reliably well in different scenarios; that is we try to find a compromise that considers all of our test cases. For example, the rank of the strategy in the average performance degradation could not be the same for any of the metrics individually or for any of the grid scenarios individually.

We analyze 4 scheduling strategies: WQ, *WQR1*, *WQR2*, *WQR3* in 400 experiments, considering 100 scenarios, and combining 20 BoT with 5 different grids.

As expected, the makespan of the schedules is improved as the number of replicas increased. If the mean task size is decreased from $\bar{p}_j = 125000$ to $\bar{p}_j = 1000$ the difference in degradations becomes neglected (Figure 2).

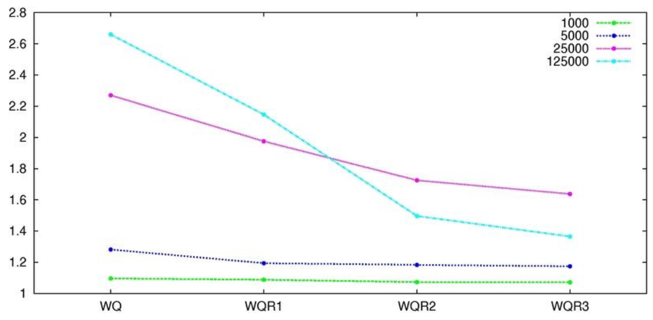


Figure 2. Mean approximation factor vs. scheduling algorithm per mean task size

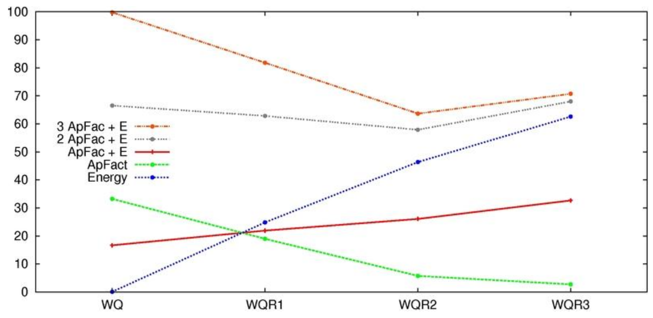


Figure 3. Mean degradation of all metrics (ρ, E^{grid}), and their weighted combinations for all test cases.

Table 3. Algorithm degradation per metric

Algorithm	ρ	E^{grid}	$\rho + E^{grid}/2$	$2\rho + E^{grid}/3$
WQ	33.24%	0.07%	16.65%	66.55%
WQR1	18.98%	24.86%	21.92%	62.82%
WQR2	5.76%	46.37%	26.07%	57.9%
WQR3	2.72%	62.60%	32.66%	68.04%

Approximation factor and energy consumption are different and conflicting performance goals. As mentioned in Section 5, there are various ways to model preferences, for instance, they can be given explicitly to specify an importance of every criterion or a relative importance between criteria.

Figure 3 shows the performance degradation of the approximation factor (ApFactor), and energy consumption (Energy) versus replication strategies *WQ*, *WQR1*, *WQR2*, *WQR3* assuming different weights of makespan criterion relatively to the energy criterion. We set the weight to 1, 2, and 3. Hence, we assume equal importance of the criteria (Energy, ApFactor), makespan importance twice

(2ApFactor+E) and three times (3ApFactor+E) as much as energy consumption.

Performance degradation of ApFactor is reduced from 33.2% with WQ to 2.7% with WQR3. Energy degradation (Energy) is increased by number of replicas from 0.06% with WQ to 62.6% with WQR3. Replication mechanism improves QoS but waists computing cycles and power, even all replicas are canceled, when a task is completed. With increasing task sizes the wasting energy becomes more considerable.

In 2ApFactor+E and 3ApFactor+E scenarios, we conclude that the compromise between QoS and energy consumption can be found using WQR2 (Figure 3, Table 3).

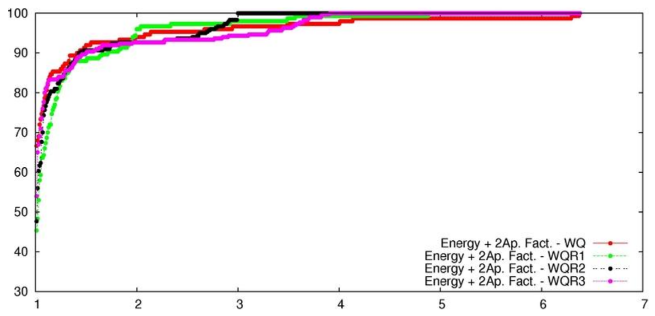


Figure 4. Performance profiles in the interval $\tau = [1..7]$

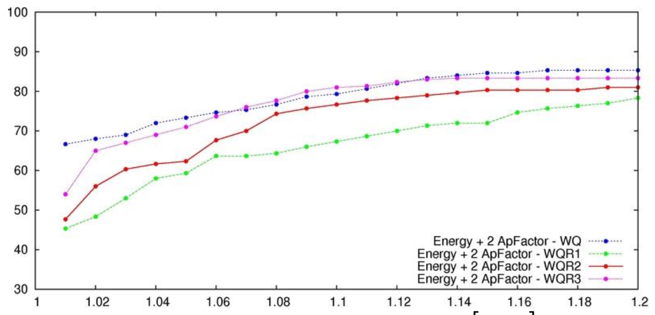


Figure 5. Performance profiles in the interval $\tau = [1...1.2]$

Figure 4 and Figure 5 show the performance profiles of the 4 strategies (2ApFactor+E with 4 replication schemes) in the interval $\tau = [1..7]$ and $\tau = [1...1.2]$, respectively. They show the provability of obtaining certain degradation in performance of the metrics (τ). Strategies with large probability $\rho(\tau)$ for smaller τ are to be preferred.

We observe that considered strategies are stable in its performance. The probability that they are the winners on a given problem within a factor of 2 of the best solution is about 0.95, and within a factor of 1.5 is about 0.90. (Figure 5). If we choose being within a factor of 1.1 as the scope of our interest, then either WQ, WQ2 and WQ3 would suffice with a probability 0.77, and WQ1 with a probability 0.65, (Figure 5). No strategy dominates.

8. Conclusions and Future Work

In this paper, we analyzed a variety of knowledge free scheduling algorithms with QoS based on mean approximation factor (ρ) and mean energy consumption (E^{grid}) with different grids and workloads. Replication strategies enhance a makespan but provide significant increase in energy consumption.

We show that by combining objectives with preferences that specify an importance of every criterion or a relative importance between criteria, a compromise between better QoS and increased energy consumption can be found.

The proposed strategies can be easily adapted to solve online scheduling problem without extra computational complexity. However, further study is required to assess their actual efficiency and effectiveness. This will be the subject of future work. Moreover, scheduling in Grid environment, where information about donated machines like speed, power efficiency, job execution historical data, etc. is available to make allocation decisions, is another important issue to be addressed. The communication latency is an additional relevant issues to be considered.

References

[1] Kurp. P..“Green Computing”. Communications of the ACM, Vol. 51, Issue 10, pp. 11-13. 2008.

[2] Sharma K. and Aggarwal S. “Energy Aware Scheduling on Desktop Grid Environment with Static Performance Prediction”. SpringSim ’09 Proceeding of the 2009 Spring Simulation Multiconference, Article No. 105. 2009.

[3] Beloglazov, A., Buyya, R., Lee, Y., C., and Zomaya, A. Y.. “A Taxonomy and Survey of Energy-Efficient Data Centers and Cloud Computing Systems”. Advances in Computers, 82, pp. 47–111. 2011.

[4] Develder C., Pickavet M., Dhoedt B. and Demeester P.. “A Power-Saving Strategy for Grids”. The Second International Conference on Networks for Grid Applications. 2008.

[5] Kim K. H., Buyya R. and Kim J.. “Power Aware Scheduling of Bag-of-Tasks Applications with Deadline Constraints on DVS-enabled Clusters”. Proceedings of the Seventh IEEE International Symposium on Cluster Computing and the Grid (CCGRID '07), pp. 541-548. 2007.

[6] Pinheiro E., Bianchini R., Carrera E, V., Heath T.. “Load Balancing and Unbalancing for Power and Performance in Cluster-Based Systems”. Proceedings of the Workshop on Compilers and Operating Systems for Low Power. 2001.

[7] Choi S., Kim H., Byun E., Baik M., Kim S., Park C. and Hwang C.. “Characterizing and Classifying Desktop Grid”. Proceedings of the Seventh IEEE International Symposium on Cluster Computing and the Grid (CCGrid'07), pp. 743-748. 2007.

[8] Cirne W., Brasileiro F., Andrade N., Costa L., Andrade A., Novaes R. and Mowbray M.. “Labs of the World, Unite!!!”. Journal of Grid Computing, Vol. 4, No. 3, pp. 225-246. 2006.

[9] Paranhos da Silva D., Cirne W. and Brasileiro F., Grande C.. “Trading Cycles for Information: Using Replication to Schedule Bag-of-Tasks Applications on Computational Grids”. Proc. of the International Conference on Parallel and Distributed Computing (Euro-Par 2003), pp. 169-180. 2003.

[10] Cunsolo V., Distefano S., Pullafito A. and Scarpa M.. “Volunteer Computing and Desktop Cloud: the Cloud@Home Paradigm”. Eighth IEEE International Symposium on Network Computing and Applications, pp. 134-139. 2009.

[11] Tsafrir, D., Etsion, Y., Feitelson, D.G. “Backfilling using system-generated predictions rather than user runtime estimates”. IEEE Trans. Parallel Distrib. Syst. 18,789–803. 2007.

[12] Ramírez-Alcaraz J., M., Tchernykh A., Yahyapour R., Schwiegelshohn U., Quezada-Pina A, González-García J. L., Hirales-Carbajal A. “Job Allocation Strategies with User Run Time Estimates for Online Scheduling in Hierarchical Grids”. J Grid Computing, Springer –Verlag, 9:95–116. 2011.

[13] Anglano C., Canonico M., Guazzone M. “The ShareGrid Peer-to-Peer Desktop Grid: Infrastructure, Applications, and Performance Evaluation”. Journal of Grid Computing, Springer, Volume 8, Number 4, 543-570. 2010

[14] Ponciano, L,, Brasileiro, F. “On the Impact of Energy-saving Strategies in Opportunistic Grids”. In: Energy Efficient Grids, Clouds and Clusters Workshop (E2GC2), 2010, Brussels, Belgium. Proceedings of the 11th ACM/IEEE International Conference on Grid Computing, 2010.

[15] DatacenterDynamics research report. Consulted online 2012. http://www.datacenterdynamics.com/research

[16] Valentini, G., Lassonde, W., Khan, S., Min-Allah, N., Madani, S., Li, J., Zhang, L., Ghani, N., Kolodziej, J., Li, H., Zomaya, A.Y., Xu, C.-Z., Balaji, P., Vishnu, A., Pinel, F., Pecero, J., Kliazovich, D., and Bouvry, P.: An overview of energy efficiency techniques in cluster computing system. Cluster Computing, doi: 10.1007/s10586-11-0171-x. 2011

[17] Dolan, E. D., Moré, J. J.: Benchmarking optimization software with performance profiles. Mathematical Programming, 91 (2) , pp 201–213 (2002).

[18] Lammie, M., Thain, D., Brenner, P.: Scheduling Grid Workloads on Multicore Clusters to Minimize Energy and Maximize Performance. 10th IEEE/ACM International Conference on IEEE Grid Computing, Banff, AB, 145–152. 2009

[19] DaCosta,G.,Gelas,J.-P.,Georgiou,Y.,Lefe`vre,L.,Org- erie, A.-C., Pierson, J.-M., Richard, O., Sharma, K.: The GREEN-NET framework: Energy efficiency in large scale distributed systems. In Procs of IEEE Int Symposium Par- allel & Distributed Processing (IPDPS 2009), Rome, 1-8. 2009

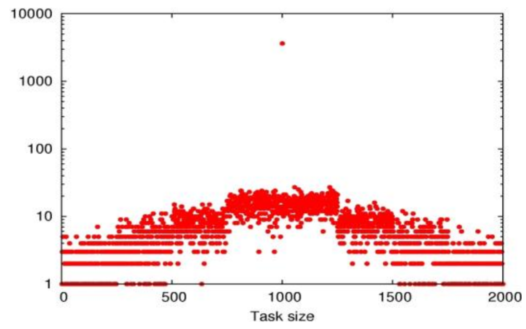
[20] Hirales-Carbajal, A., Tchernykh, A., Roblitz, T., Yahyapour, R.: A grid simulation framework to study advance scheduling strategies for complex workflow applications. In: IEEE International Symposium on Parallel Distributed Processing, Workshops and Phd Forum (IPDPSW), pp. 1–8 2010.

[21] Rodrigues, R. and Druschel, P.: Peer-to-peer systems. In: Commun. ACM. Vol 53. Num 53, pp. 72-82. 2010.

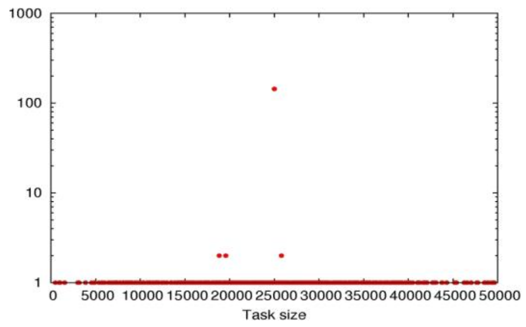
[22] Berral, J. Ll., Goiri, I., Nou, R., Juliá, F., Guitart, J., Gavaldá, R., Torres, J.: Towards energy-aware scheduling in data centers using machine learning. In. proceedings of the 1st International Conference on Energy-Efficient Computing and Networking. Vol 53. Num 53, pp. 215-224. 2010.

[23] Dolan, E. D., Moré, J. J., Munson, T. S.: Optimality measures for performance profiles. Siam Journal on Optimization, 16, pp 891–909 (2006).

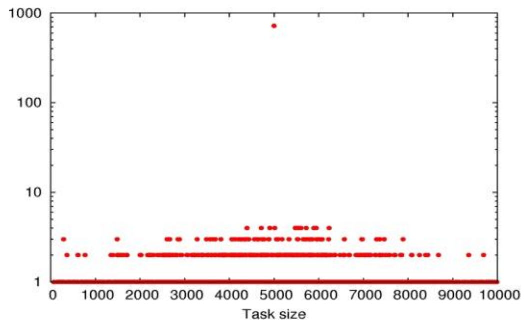
Appendix



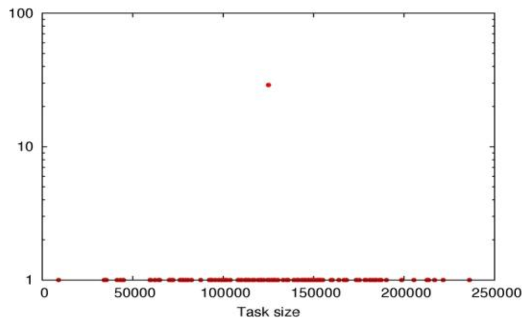
$\bar{p}_j = 1000$



$\bar{p}_j = 25000$



$\bar{p}_j = 5000$



$\bar{p}_j = 125000$

Figure A. Histograms of BoT with $\bar{p}_j = 1000$, $\bar{p}_j = 5000$, $\bar{p}_j = 25000$, $\bar{p}_j = 125000$