Performance and Energy Aware Scheduling Simulator for High-Performance Computing

César Gómez-Martín (CénitS)[1], Miguel A. Vega-Rodríguez (University of Extremadura)[2], José-Luis González-Sánchez (CénitS)[1], Javier Corral-García (CénitS)[1] and David Cortés-Polo (CénitS)[1]

[1]CénitS (Research, Technological Innovation and Supercomputing Center of Extremadura) - N-521, Km. 41,8 10071 Cáceres (Spain)
{cesar.gomez,joseluis.gonzalez,javier.corral,david.cortes}@cenits.es
[2]University of Extremadura - Escuela Politécnica de Cáceres, Avda. Universidad s/n 10003 Cáceres (Spain)
mavega@unex.es

Abstract. Job scheduling is a very complex task that influences the performance of High-Performance Computing (HPC) infrastructures, thus, a simulator of real computing environments for an accurate evaluation of scheduling and resource selection policies helps ICT and data center managers to make decisions with a solid experimental basis. There are several simulators that try to address performance and somehow estimate energy consumption, but there are none in which the energy model is based on benchmark data that have been countersigned by independent bodies such as the Standard Performance Evaluation Corporation (SPEC), this is the reason why we have implemented a Performance and Energy Aware Scheduling (PEAS) Simulator for HPC. In order to validate the simulator and understand how different computing workloads behave depending on scheduling policies, several user-centric evaluation metrics have been evaluated and compared with previous studies.

1 Introduction

Today job scheduling is still a very complex task that may have a significant influence on the performance of computing infrastructure, thus, a good job scheduling system is very important to reduce operating costs. A study of actual workloads can significantly increase the energy efficiency of IT infrastructure. For more than half a century the scientific computing community has mainly focused on system performance while forgetting other important aspects like energy efficiency. Nowadays, with the arrival of embedded computers, the boom in battery dependent devices, higher energy costs and global warming, energy efficiency is becoming more important not only for hardware vendors, but also for ICT and data center managers, and green computing is no longer seen as an oxymoron. Moreover, the notion of power awareness or low power supercomputing is new [1], while extending battery life is the main goal in mobile devices, the main motivation of saving energy in data centers is to reduce operating costs, that is the reason why new low-power Atom/ARM based servers are being introduced in supercomputing and regular data centers. When dealing with power savings among mobile, embedded and High-Performance Computing (HPC) applications, there are some differences and most of them are related to the need to address real-time operations in mobile environments or meeting wall-clock time requirements in some HPC scientific applications. There have been several works that study the impact of DVFS (Dynamic Voltage and Frequency Scaling) combined with a power-aware algorithm to save energy [2, 3], and some others about how to balance and save energy in clusters with virtual machines [4]. The innovation of the simulator presented in this paper is the inclusion of a Power and Performance benchmark, from the Standard Performance Evaluation Corporation (SPEC), to model the energy consumption and the performance of a cluster of servers, or a supercomputer, when running a set of jobs with a job scheduler.

1.1 Related Work

Application capabilities and data volumes are increasing exponentially, this leads to the need for more computing capacity and better management of HPC environments, so several business solutions use the above job data for intelligent and complex workload scheduling:

- IBM Platform LSF (Load Sharing Facility): a workload management platform for demanding and distributed HPC environments.
- SLURM is an open-source resource manager that is becoming popular because it can work with heterogeneous clusters of all sizes.
- PBS (Portable Batch System): originally developed for NASA to allocate computational tasks. There are various open implementations, OpenPBS and TORQUE, together with a commercial version called PBS Pro.
- Oracle Grid Engine: a distributed resource management (DRM) system that manages the distribution of user workloads to available computer resources.
- HTCondor: is an open-source computing framework for coarse-grained parallelization of computationally intensive tasks.

In [5] Krallmann, Schwiegelshohn, and Yahyapour defined three categories for job data as an approach to designing a job scheduling system:

- User Data: used to determine job priorities.
- Resource Requests: these data usually specify the number and architecture of the processors, the amount of memory, an estimation of the running time, etc.
- Scheduling Objectives: data that help the scheduler to generate "good" schedules.

According to the authors, a scheduling system should be divided into three parts: scheduling policy, objective function and scheduling algorithms. Other authors introduce a fourth decision making element called resource selection policy [6] to allocate jobs depending on the resource requirements or limitations.

There are simulators like MuPSiE [7] that follow the core principles of [5] and some others that also use a separate resource selection policy component like Alvio [6] and Kento-sim [8]. Other works propose energy management in multiprocessor environments with machine learning techniques such as [9].

1.2 Contributions

To properly manage a data center in general and a supercomputer in particular, in CénitS, along with the University of Extremadura, we are currently developing a Performance and Energy Aware Scheduling (PEAS) Simulator that will help us in decision making for the configuration of our workload management platform. Furthermore, with the simulator we intend to develop new resource selection and scheduling policies for better management of compute resources and energy consumption. In this paper we present the PEAS Simulator as an evaluation tool for system analysts and data center managers to define scheduling policies for their business solutions. The PEAS simulator has the following improvements over existing simulators:

- Implementation of a configurable resource selector able to meet certain constraints (energy consumption, wall clock time, performance, etc.).
- Use of actual power and performance standard benchmarks of current computing servers that generate new metrics and will allow new resource selection policies to be added in the future.
- Inclusion of updated workloads and benchmarking results in the simulator by simply adding standard files from the Parallel Workloads Archive and SPEC Power and Performance repository.

An early version of PEAS Simulator has been used to simulate real workloads from several supercomputing centers using industry accepted and widely recognized scheduling and resource selection policies.

In the remainder of the paper, we briefly describe the PEAS Simulator architecture in Section 2. Thereafter, in Section 3, we explain how the jobs are processed by the simulator to calculate power and time metrics and evaluate how "well" the scheduler behaves. Section 4 presents simulation results of some real computing center workloads that are used to validate the PEAS Simulator. In Section 5 we introduce the expected improvements of the simulator in upcoming versions. Finally, in Section 6, we outline the conclusions of this paper.

2 Architecture of the simulator

The PEAS Simulator is an event-driven job scheduler simulator that is meant to help in the evaluation and optimal utilization of hardware and energy resources of HPC environments. In Figure 1 the architecture of the simulator is sketched:

- Prior to the start of the simulation, a file containing the jobs workload in SWF format and a CSV file with the cluster configuration are processed.

- After the workload and the hardware resources are loaded into the simulator, the scheduling and resource selection policies are read from a configuration file.
- Once all the configuration files are properly processed, the scheduler starts to decide which jobs are processed (Policy Scheduler) and, with the help of the Resource Selector, assigns free resources to the highest priority request. If the job requests more resources than the total number of resources provided by the whole cluster, the request is not accepted by the scheduler (Unaccepted Jobs). If at a particular moment there are not enough free resources, the job is queued (Queued Jobs) and waits until enough resources are available.
- Finally, when all the jobs are finished, an output file containing all the evaluation metrics of every job and the whole schedule is generated. It is also possible to generate a log file that includes debugging information.

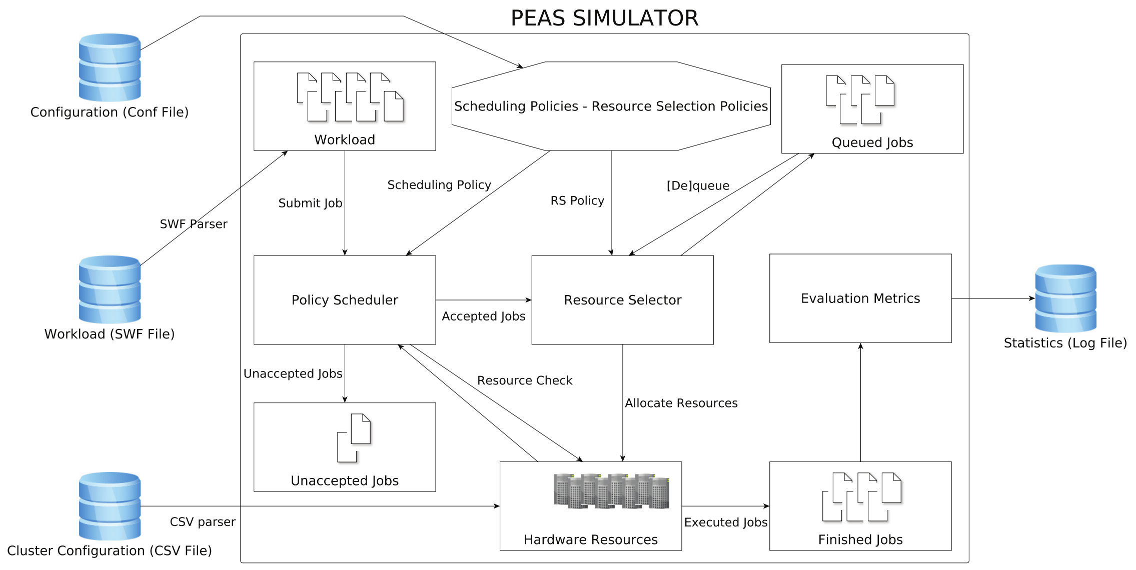


Fig. 1. PEAS Simulator Architecture

2.1 Workload modeling

A job can be defined by the following parameters:

- Submit time in seconds. Usually the first submitted job has a submit time of 0, meanwhile the rest of the jobs are sorted by ascending submittal times.
- Estimated duration of the job. This field is used for the user runtime estimate (or upper bound) and it is useful for backfilling policies.
- Actual duration of the job. The wall clock time the job was running (end time minus start time).
- Requested resources (number of processing cores, amount of main memory, etc.).

– Some other parameters of interest for some scheduling policies.

Today most supercomputing centers have a resource management software that is able to easily generate job traces, this is very important for an accurate study of scheduling policies because it is not easy to randomly generate traces for "real world" workloads. In order to face the latter problem, the parallel workloads that are being used to test the PEAS simulator have been extracted from the Parallel Workload Archive [10], which has a wide variety of real traces from supercomputing centers of very diverse nature. The standard workload format (SWF) was proposed by David Talby and refined through discussions by Dror Feitelson, James Patton Jones, and others [11]. We decided to use this format because it is very easy to parse, since every job is well defined and contains all the data that is needed to properly simulate a real HPC environment. In the simulator we have developed a parser for version 2.2, which is the last published version. In table 1 there is a real example of some parameters from a workload of the RIKEN Integrated Cluster of Clusters, RIKEN is an independent scientific research and technology institution of the Japanese government.

Table 1. Example Workload - RICC: RIKEN Integrated Cluster of Cluster, Oct 6, 2011

Job #	Submit Time	Runtime Estimate	Runtime	# Used CPUs	...
1	0	14400	222	80	...
2	1136	259200	244682	128	...
3	1160	259200	249628	128	...
4	1877	259200	259209	128	...
5	1903	259200	211161	128	...
6	1920	86400	78509	128	...
7	1920	86400	78391	128	...
8	1920	86400	77206	128	...
9	1920	86400	79139	128	...
10	1920	86400	77584	128	...
11	...	...	...	...	...

2.2 Power and Performance modeling

The power a microprocessor consumes and dissipates varies depending on the manufacturer, moreover, power by itself is not a measurement of efficiency, a compromise between performance and power consumption is better for defining energy efficiency. A system which consumes low levels of power but does not perform well will take longer to perform a task, and may ultimately consume more energy, thus, a more efficient server is the one that has the best performance per watt [12]. When comparing power specifications it is important to use metrics that are able to measure the same parameters. Intel and AMD have designed their own power specifications:

– Intel Thermal Design Power (TDP): is the maximum power a processor can draw for a thermally significant period while running commercially useful software.

– AMD Average CPU Power (ACP): is the average (geometric mean) power a processor was measured to dissipate while running a collection of four different benchmarks: TPC Benchmark*-C, SPECcpu, SPECjbb and STREAM.

Intel and AMD agree on the fact that it is not a good idea to measure power consumption by only looking at the spec sheets of different components and adding the totals together, because these only report the maximum power consumption. The best way to calculate the power consumed by "real world" workloads is to use a power meter [13]. In order to evaluate not only performance but also power usage, SPEC designed the SPEC Power and Performance benchmark which is the first industry standard that evaluates the power and performance characteristics of volume server class and multi-node class computers [14]. In the PEAS Simulator we have implemented a CSV parser that is able to process the files provided by SPEC and complies with version 1.12 of the SPEC Power and Performance benchmark. In table 2 the most relevant fields of a cluster configuration file in CSV format are shown.

Table 2. Example of Cluster configuration file

Vendor	Processor	#Chips	#Cores/Chip	RAM (GB)	OPS@100%Load	Watts@100%Load	...
ASUS	Xeon X3360	1	4	4.00	165064	118	...
ASUS	Xeon L5420	2	4	8.00	270621	170	...
ASUS	Xeon L5430	2	4	8.00	278927	173	...
Acer	Xeon X3470	1	4	8.00	309401	125	...
Acer	Xeon E3-1260L	1	4	8.0	295443	58	...
Acer	Xeon X5670	2	6	12.0	898544	267	...
Acer	Xeon X5670	2	6	12.0	897310	265	...
Acer	Xeon X3470	1	4	8.00	308318	124	...
Acer	Xeon X5670	2	6	12.0	890144	272	...

2.3 Scheduling Policy component

This component is in charge of storing all the jobs in a list-like structure that is ordered by arrival time. First come first served (FCFS) scheduling policies are the most common in queuing systems, they are easy to understand from a user point of view, although they are probably not the best option for a system administrator. Traditional FCFS is only used for experiments or studies, it is not a real alternative because it leaves a lot of resources idle. This issue is solved in modern job scheduling system with the inclusion of backfilling variants. With a backfilling mechanism all the idle resources may be utilized by lower priority jobs. In the current version of the PEAS Simulator we have implemented three scheduling policies:

– FCFS: first come first served is a well known scheduling policy that generates fair schedules. It is very inefficient due to fragmentation but it is easy to understand; jobs that arrive later are started later.

– FCFS Conservative Backfilling: this variant moves jobs forward only if all the previously queued jobs are not delayed. It does not affect any queued job, hence it is known as the conservative version [15].
– FCFS EASY Backfilling: EASY stands for "Extensible Argonne Scheduling System", it was developed for the IBM SP2 supercomputer and is also called Aggressive Backfilling because it allows short jobs to move forward if they do not delay only the first job of the queue, although the rest of the jobs ahead of it in the queue may be delayed [16].

2.4 Resource Selection Policy component

The resource selection policy component decides the nodes and processors among which a job must be allocated. Depending on the number of cores, memory, energy or power constraints, a job would be allocated in a different server. We have implemented First Fit Policy (FFP) in the current version of the PEAS Simulator. It is the simplest and most widely used selection policy. FFP finds the first n idle processors that meet the selected job constraints and allocates it to start the execution. Although in the first version of the simulator there is only one resource selection policy, this component has been designed to be easily extended in the future with new policies or algorithms.

3 Methodology

Since not all the processors and computing servers have the same performance (operations per second (OPS)), in order to compute a more reliable runtime for our simulation (SimulatedRunTime), we have to calculate the number of operations that correspond to every job of the workload. For that purpose, the actual time of the job, which can be obtained from the SWF file, is assumed to be the time spent by the job in a fictitious average processor. The fictitious processor is calculated as the average processor of all the servers that are inside the cluster configuration file at a 100% load (see subsection 2.2).

$$OPS_{CPU_{Average}} = \frac{\sum_{i=1}^{n} OPS_{CPU_i}}{n} \tag{1}$$

where:

– OPS_{CPU_i} is the number of operations per second at 100% load in CPU_i.
– n is the total number of CPUs of the cluster.

Based on the actual runtime of a job, the simulated runtime of the same job on the corresponding processor of the cluster is calculated as follows:

$$SimulatedRunTime = (\max_{i=1...AllocCPUs} \frac{OPS_{CPU_{Average}}}{OPS_{CPU_i}}) * Runtime \tag{2}$$

where:

– $Runtime$ is the actual runtime of the job.
– OPS_{CPU_i} is the number of operations per second at 100% load in CPU_i.
– $OPS_{CPU_{Average}}$ is the number of operations per second at 100% load in $CPU_{Average}$. (See equation 1).
– $AllocCPUs$ is the number of allocated CPUs in which the job has been run.

3.1 Performance metrics

To evaluate how "good" the different policies are two user-centric performance metrics are calculated by the simulator:

– The average waiting time of the jobs is calculated as follows:

$$AWT = \frac{\sum_{i=1}^{n} t_i^w}{n} \tag{3}$$

where:
- t_i^w is the waiting time of Job_i.
- n is the total number of jobs that have been accepted by the scheduler.

– The average response time can be also calculated:

$$ART = \frac{\sum_{i=1}^{n} t_i^r}{n} \tag{4}$$

where:
- t_i^r is the response time of Job_i.
- n is the total number of jobs that have been accepted by the scheduler.

3.2 Energy metrics

Finally, to calculate the power consumption of every job, the energy needed by all the working processors to accomplish a given job is calculated with the following formula:

$$PowerConsumption_{Job} = \sum_{i=1}^{AllocCPUs} Watts@100\%Load_{CPU_i} * SimulatedRunTime \tag{5}$$

where:

– $Watts@100\%Load_{CPU_i}$ is the energy consumption of CPU_i at 100% load. (See table 2).
– $AllocCPUs$ is the number of allocated CPUs in which the job has been run.

Table 3. Performance and energy metrics for HPC2N, LLNL and RICC (100 jobs)

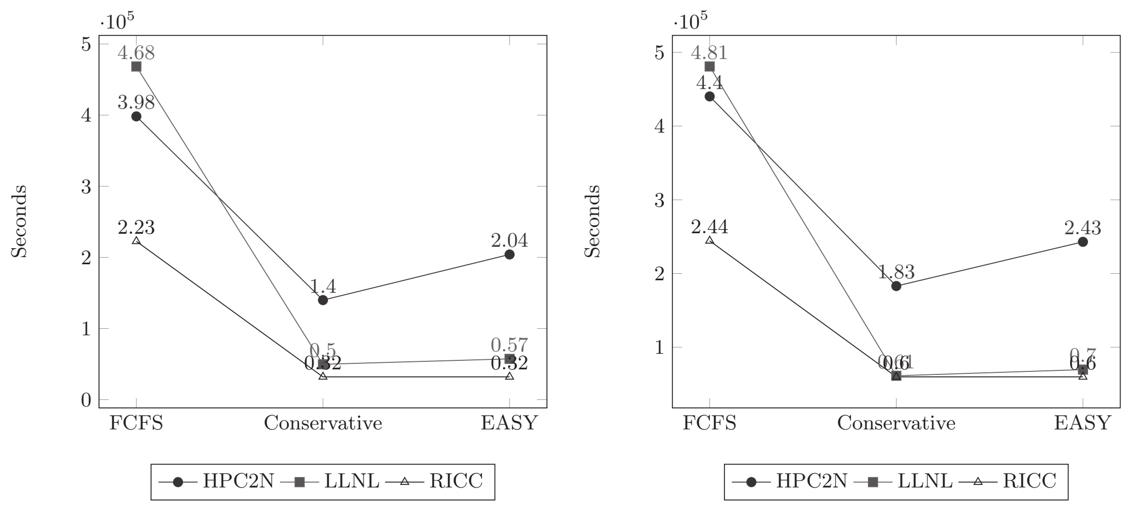
	FCFS			Conservative			EASY		
	HPC2N	LLNL	RICC	HPC2N	LLNL	RICC	HPC2N	LLNL	RICC
AWT (seconds)	398258	468456	222651	139969	49734	32029	204112	57325	32029
ART (seconds)	440243	480897	244332	183034	61326	59830	243002	69698	59830
Power Consumption (KWh)	269.417	214.058	34.0465	249.363	202.647	38.2591	237.208	205.421	38.2591

4 Experimental Results

In order to validate the PEAS Simulator, three workloads from large scale parallel systems in production have been tested. The configuration of the cluster can be seen in table 2. Since most of the logs contain several years of submitted jobs, only the first hundred of each logfile have been used for the validation process.

– HPC2N: is the workload of the High-Performance Computing Center North (HPC2N) in Sweden.
– LLNL: is the workload of a large Linux cluster called Thunder installed at Lawrence Livermore National Lab (LLNL).
– RICC: contains several months of jobs from the RIKEN Integrated Cluster of Clusters (RICC).

The results can be observed in Table 3.



(a) Average Waiting Time (b) Average Response Time

Fig. 2. Average Waiting Time & Average Response Time

Figure 2.a shows the average waiting time (AWT) for FCFS, FCFS Conservative Backfilling (Conservative) and FCFS EASY Backfilling (EASY). Most of the jobs within the three workloads are wide jobs, i.e. they need a lot of hardware resources, so it is hard for them to easily find enough idle processors. The reason why there is a better AWT in conservative backfilling is because the scheduler guarantees them a start time avoiding starvation. In Figure 2.b the average response times (ART) for FCFS, Conservative and EASY are compared. There is again a better ART in conservative backfilling because of long wide jobs. EASY backfilling does not give a reservation for them, so more jobs can backfill ahead. There is no clear advantage between EASY and conservative backfilling, it depends on the characteristics of the workload. While EASY clearly benefits long narrow jobs, there is no clear advantage for long wide jobs. Conservative backfilling provides reservation while EASY offers better backfilling opportunities due to fewer blockages in the schedule [17].

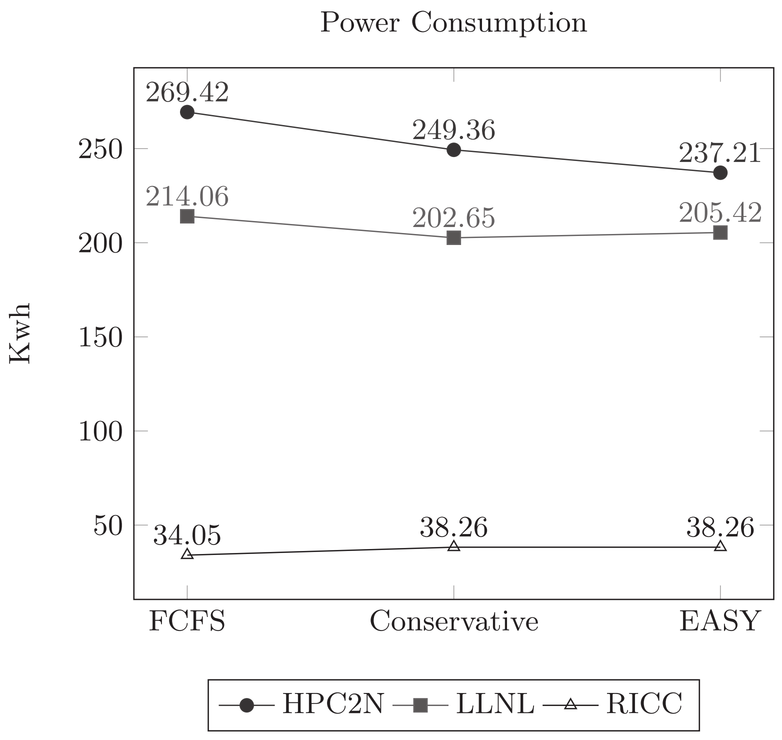


Fig. 3. Power Consumption

As expected, due to the resource selection policy (first-fit), energy consumption does not follow a clear pattern because it greatly depends on which processors are free when a job starts running (see Figure 3). However, this metric is especially important for future implementations of hardware selection policies that will try to reduce power consumption with optimization algorithms.

5 Future Work

Future contributions to the simulator are expected, including the following:

– Implementation of additional user-centric performance metrics weighted by job width [18]: average response time weighted by job width and average slowdown weighted by job area.
– Implementation of classic scheduling policies for experimental purposes with different sorting criteria such as: shortest job first (SJF), longest job first (LJF) and others.
– Multiobjective optimization algorithms will be used for a more "intelligent" scheduling policy.
– In order to quantify the impacts and benefits of modern power management functions, the SPEC Power and Performance benchmark also provides power consumption and the number of operations per second for a range of eleven throughput levels, from idle to 100%. We currently do not use graduated measurement intervals but we plan to use them to find a good compromise between power and performance.

6 Conclusions

Although green computing is somehow still seen as an oxymoron for HPC center managers, most of them are aware of the importance of reducing operational costs even though performance is still the main goal for HPC. With the PEAS Simulator we intend to help in the management of data center workloads trying not only to achieve good response times but also to consume as little energy as possible. There is a wide field of study for scheduling and resource selection policies combined with artificial intelligent and multicriteria optimization and the PEAS Simulator has been structured and designed to implement new algorithms to address some of the unknowns of system analysts and administrators. In this paper we present a tested early version of PEAS Simulator, with some of the scheduling policies that are implemented by modern queuing and scheduling systems. The main contribution of the PEAS Simulator is the implementation of an innovative Power and Performance aware component that is able to model the actual energy consumption of real computing center workloads. Further versions of the PEAS Simulator are expected to include improvements as well as possibilities for helping ICT managers manage computing infrastructures, and will include artificial intelligence and multi-objective optimization algorithms for better utilization of hardware resources.

Acknowledgements

This research is part financed by the ERDF Fund Programme: Extremadura Operational Programme 2007-2013, Development of the Knowledge Economy (R&D&I: Information Society and ICTs), Research and Technology Development (R&TD) Activities in Research Centres.

References

1. Hsu, C. H., & Feng, W. C. (2005, November). A power-aware run-time system for high-performance computing. In Proceedings of the 2005 ACM/IEEE conference on Supercomputing (p. 1). IEEE Computer Society.
2. Choi, K., Soma, R., & Pedram, M. (2004, August). Dynamic voltage and frequency scaling based on workload decomposition. In Proceedings of the 2004 international symposium on Low power electronics and design (pp. 174-179). ACM.
3. Chen, J. J., & Kuo, C. F. (2007, August). Energy-efficient scheduling for real-time systems on dynamic voltage scaling (DVS) platforms. In Embedded and Real-Time Computing Systems and Applications, 2007. RTCSA 2007. 13th IEEE International Conference on (pp. 28-38). IEEE.
4. Hu, L., Jin, H., Liao, X., Xiong, X., & Liu, H. (2008, September). Magnet: A novel scheduling policy for power reduction in cluster with virtual machines. In Cluster Computing, 2008 IEEE International Conference on (pp. 13-22). IEEE.
5. Krallmann, J., Schwiegelshohn, U., & Yahyapour, R. (1999, January). On the design and evaluation of job scheduling algorithms. In Job Scheduling Strategies for Parallel Processing (pp. 17-42). Springer Berlin Heidelberg.
6. Guim, F., Corbalan, J., & Labarta, J. (2007, October). Modeling the impact of resource sharing in backfilling policies using the alvio simulator. In Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, 2007. MASCOTS'07. 15th International Symposium on (pp. 145-150). IEEE.
7. Heine, F., Hovestadt, M., Kao, O., & Streit, A. (2005). On the impact of reservations from the grid on planning-based resource management. In Computational ScienceICCS 2005 (pp. 155-162). Springer Berlin Heidelberg.
8. Guim, F., Rodero, I., Corbalan, J., & Parashar, M. (2010, September). Enabling gpu and many-core systems in heterogeneous hpc environments using memory considerations. In High Performance Computing and Communications (HPCC), 2010 12th IEEE International Conference on (pp. 146-155). IEEE.
9. Berral, J. L., Goiri, Í., Nou, R., Julià, F., Guitart, J., Gavaldà, R., & Torres, J. (2010, April). Towards energy-aware scheduling in data centers using machine learning. In Proceedings of the 1st International Conference on energy-Efficient Computing and Networking (pp. 215-224). ACM.
10. Feitelson, D. G., Tsafrir, D., & Krakov, D. (2012). Experience with the parallel workloads archive. Technical Report 2012-6. The Hebrew University of Jerusalem.
11. Chapin, S. J., Cirne, W., Feitelson, D. G., Jones, J. P., Leutenegger, S. T., Schwiegelshohn, U., ... & Talby, D. (1999, January). Benchmarks and standards for the evaluation of parallel job schedulers. In Job Scheduling Strategies for Parallel Processing (pp. 67-90). Springer Berlin Heidelberg.
12. Huck, S. (2011). Measuring Processor Power: TDP vs. ACP. White Paper, Revision 1.1. Intel.
13. AMD. ACP - The truth about power consumption starts here. White Paper: Power Consumption. AMD.
14. Poess, M., Nambiar, R. O., Vaid, K., Stephens Jr, J. M., Huppler, K., & Haines, E. (2010, April). Energy benchmarks: a detailed analysis. In Proceedings of the 1st International Conference on Energy-Efficient Computing and Networking (pp. 131-140). ACM.
15. Lifka, D. A. (1995, January). The anl/ibm sp scheduling system. In Job Scheduling Strategies for Parallel Processing (pp. 295-303). Springer Berlin Heidelberg.

16. Mu'alem, A. W., & Feitelson, D. G. (2001). Utilization, predictability, workloads, and user runtime estimates in scheduling the IBM SP2 with backfilling. Parallel and Distributed Systems, IEEE Transactions on, 12(6), 529-543.
17. Srinivasan, S., Kettimuthu, R., Subramani, V., & Sadayappan, P. (2002). Characterization of backfilling strategies for parallel job scheduling. In Parallel Processing Workshops, 2002. Proceedings. International Conference on (pp. 514-519). IEEE.
18. Streit, A. (2003). Self-tuning job scheduling strategies for the resource management of HPC systems and computational grids (Doctoral dissertation, PhD thesis, Faculty of Computer Science, Electrical Engineering and Mathematics, University Paderborn).