Characterizing, Modeling, and Accurately Simulating Power and Energy Consumption of I/O-intensive Scientific Workflows

Rafael Ferreira da Silva[a,*], Henri Casanova[b,**], Anne-Cécile Orgerie[c], Ryan Tanaka[a,b], Ewa Deelman[a], Frédéric Suter[d,e]

[a] *University of Southern California, Information Sciences Institute, Marina del Rey, CA, USA*
[b] *Information and Computer Sciences, University of Hawaii, Honolulu, HI, USA*
[c] *Univ Rennes, Inria, CNRS, IRISA, Rennes, France*
[d] *IN2P3 Computing Center, CNRS, Villeurbanne, France*
[e] *Inria, LIP, ENS Lyon, Lyon, France*

Abstract

While distributed computing infrastructures can provide infrastructure-level techniques for managing energy consumption, application-level energy consumption models have also been developed to support energy-efficient scheduling and resource provisioning algorithms. In this work, we analyze the accuracy of a widely-used application-level model that has been developed and used in the context of scientific workflow executions. To this end, we profile two production scientific workflows on a distributed platform instrumented with power meters. We then conduct an analysis of power and energy consumption measurements. This analysis shows that power consumption is not linearly related to CPU utilization and that I/O operations significantly impact power, and thus energy, consumption. We then propose a power consumption model that accounts for I/O operations, including the impact of waiting for these operations to complete, and for concurrent task executions on multi-socket, multi-core compute nodes. We implement our proposed model as part of a simulator that allows us to draw direct comparisons between real-world and modeled power and energy consumption. We find that our model has high accuracy when compared to real-world executions. Furthermore, our model improves accuracy by about two orders of magnitude when compared to the traditional models used in the energy-efficient workflow scheduling literature.

Keywords: Scientific workflows, Energy-aware computing, Workflow profiling, Workflow scheduling.

1. Introduction

Computational workloads that require from a few hours to a few months of execution are commonplace in scientific simulations. These simulations can comprise many computational tasks, which are often I/O-intensive (i.e., a non-negligible and often major portion of the execution is spent doing I/O operations rather than computation), with some dependency structure. This is why many scientists today formulate their computational problems as scientific workflows [1]. To obtain simulation results within acceptable time-frames, large scientific workloads are executed on distributed computing infrastructures such as grids and clouds [2]. The need to manage energy consumption across the entire suite of computation, storage, and communication technology has received significant attention in the last few years [3, 4]. As a result, large data-centers have developed techniques for managing cooling and energy usage at the infrastructure level. Concurrently, researchers have investigated application-level techniques and algorithms to enable energy-efficient executions [5]. In the context of scientific workflows, researchers have proposed a range of energy-aware workflow task scheduling or resource provisioning algorithms [6, 7, 8, 9, 10, 11, 12, 13, 14, 15]. Results therein are obtained based on a model of power consumption that is easy to instantiate but that makes strong assumptions: power consumption is considered to be linearly correlated with CPU utilization, and equally divided among virtual machines or CPU cores within a computational node. If dynamic voltage and frequency scaling (DVFS) is enabled, power consumption is also assumed to be estimated reasonably well as a cubic function of the frequency. An interesting question is whether this model is accurate in practice, and whether it can be applied to I/O-intensive workflow executions.

Our broad objective in this work is to characterize the energy consumption behavior of complex workflow applications that execute on distributed platforms. We profile real scientific workflow applications on a platform that comprises multi-socket, multi-core compute nodes equipped with power meters. We select two widely used scientific workflows, each of which has many I/O-intensive tasks. We conduct a comprehensive analysis of the power and energy consumption of

[*] Corresponding address: USC Information Sciences Institute, 4676 Admiralty Way Suite 1001, Marina del Rey, CA, USA, 90292

[**] Corresponding address: University of Hawaii Information and Computer Sciences, POST Building, Rm 317, 1680 East-West Road, Honolulu, HI, USA, 96822

Email addresses: `rafsilva@isi.edu` (Rafael Ferreira da Silva), `henric@hawaii.edu` (Henri Casanova), `anne-cecile.orgerie@irisa.fr` (Anne-Cécile Orgerie), `tanaka@isi.edu` (Ryan Tanaka), `deelman@isi.edu` (Ewa Deelman), `frederic.suter@cc.in2p3.fr` (Frédéric Suter) the workflows' executions. Specifically, we collect power consumption measurements for both solitary and concurrent task executions. We evaluate the performance impact of concurrent task executions and the influence of I/O operations on energy usage. We apply correlation statistics and principal component analysis to identify statistical relationships between power (and energy) consumption and the target parameters (CPU utilization and the quantity of I/O operations). Via this analysis, we quantify the accuracy of power consumption models commonly used in the energy-efficient workflow scheduling literature. We then propose a more accurate power consumption model. A key requirement for this model is that it can be instantiated using metrics measured at the application-level, i.e., by exploiting profiling data collected by existing production workflow management systems. More specifically, this work makes the following contributions[1]:

1. The power and energy consumption profiles of two real I/O-intensive scientific workflow applications;
2. A comprehensive analysis of these profiles with respect to resource utilization and I/O operations;
3. An evaluation of the accuracy of the power model that is widely used in the workflow scheduling literature;
4. A power consumption model for I/O-intensive workflows that accounts for the allocation of cores to sockets, CPU utilization, and I/O operations; this model is platform-specific but application-agnostic; and
5. An experimental evaluation of the proposed model that shows that it can produce nearly accurate energy consumption estimates, with improvements over traditional models by almost two orders of magnitude.

The paper is organized as follows. Section 2 describes the scientific workflow applications used in this work and presents execution and energy profiles of tasks in these workflows. Section 3 presents analysis of these profiles, and Section 4 analyses how concurrent task executions impact power and energy consumption. Section 5 describes our proposed power model for I/O-intensive applications and an evaluation of its accuracy. Section 6 discusses opportunities and practical challenges for applying application-level energy reduction approaches. Section 7 reviews related work and Section 8 concludes the paper with a summary of results and perspectives on future work.

2. Workflow Characterization

The analysis presented in this work is based on the execution of two production scientific workflow applications on the Grid'5000 [17] platform. Grid'5000 is a testbed for experiment-driven research, which provides resource isolation and advanced monitoring and measurement features for the collection of power consumption traces.

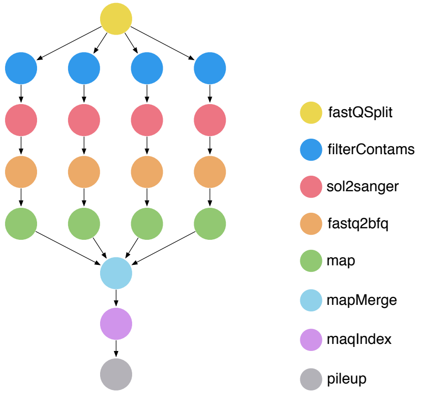

Figure 1: Epigenomics workflow.

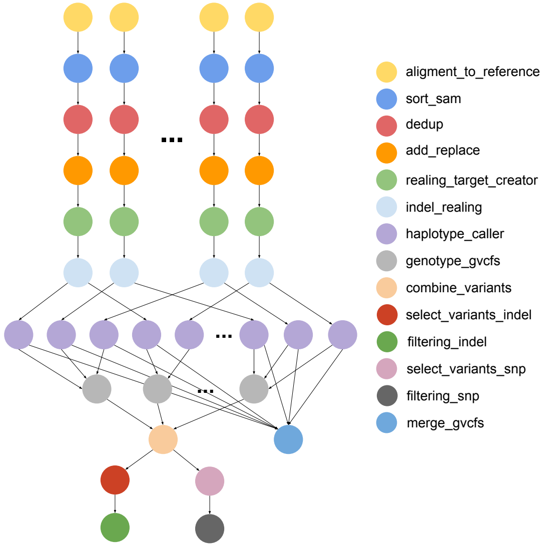

Figure 2: SoyKB workflow.

2.1. Scientific Workflows

A scientific workflow describes a set of computational tasks and the dependences between them. In many cases, workflows can be described as a directed acyclic graph (DAG), where the vertices represent tasks and the edges represent data or control dependencies [18, 19, 20]. In this work, we consider these two I/O-intensive workflows:

- *Epigenomics [21]* – A bioinformatics workflow that maps the epigenetic state of human cells on a genome-wide scale by processing multiple sets of genome sequences in parallel (Fig. 1). These sequences are split into subsets, the subsets are filtered to remove contaminants, reformatted, then mapped to a reference genome. The mapped sequences are finally merged and indexed for later analysis. We consider an Epigenomics instance with 577 tasks.
- *SoyKB [22]* – A bioinformatics workflow that resequences soybean germplasm lines selected for desirable traits such as oil, protein, soybean cyst nematode resistance, stress resistance, and root system architecture. The

[1] A preliminary version of this paper appears in the proceedings of the 2019 International Conference on Computational Science (ICCS) [16].

workflow (Fig. 2) implements a Single Nucleotide Polymorphism (SNP) and insertion/deletion (indel) identification and analysis pipeline using the GATK haplotype caller and a soybean reference genome. The workflow analyzes samples in parallel to align them to the reference genome, to de-duplicate the data, to identify indels and SNPs, and to merge and filter the results. We consider a SoyKB instance with 676 tasks.

We have selected these two workflow applications as use-cases because many of their tasks are I/O-intensive, and because their tasks exhibit a range of computation-to-I/O ratios, as seen by the diversity of their CPU usages (Table 1). Another reason for selecting these workflows is that they include both synchronous and asynchronous operations. For instance, most I/O read operations performed by tasks in our two workflows are synchronous (with exception of `pileup` and `*_snp` tasks, respectively), while most I/O write operations are asynchronous (except for `align_reference`). Overall, we claim that Epigenomics and SoyKB are good exemplars of I/O-intensive workflow applications as they include a diverse mix of I/O-intensive tasks and these tasks use both synchronous and asynchronous I/O operations. For instance, to read/write about 500MB, `fastqSplit` takes around 1.5s in average.

2.2. Workflow Execution Profile

We profiled these workflows when executed with Pegasus [18], a state-of-the-art workflow management system. Pegasus bridges the scientific domain and the execution environment by automatically mapping high-level, abstract workflow descriptions onto distributed resources. It manages data on behalf of the user: it infers the required data transfers, registers data into catalogs, and captures performance information while maintaining a common user interface for workflow submission. Pegasus obtains and logs fine-grained profiling data such as I/O operations, runtime, memory usage, and CPU utilization [23]. In this paper, we leverage these profiling data to derive a power model for I/O-intensive workflows. This model, which is presented in Section 5, accounts not only for CPU usage but also for cores/sockets allocation schemes and for I/O usage. A key aspect of our approach is that the model is instantiated solely based on application-level profiling data. This data is already obtained at runtime by production workflow management systems (in this work we use Pegasus), and thus our model can be instantiated and used by these systems without any modification to their implementation. This is by contrast with other, possibly more sophisticated models, that would require low-level hardware and operating system information that is not typically available to workflow management systems (see discussion in Section 7).

The workflows were executed on two clusters at the Grid'5000 Lyon site (*taurus* and *orion*), which are instrumented at the node level with power meters. For each cluster, we used a single node to run the workflow tasks and collect power measurements. Although time-intensive, this approach allows us to collect non-biased measurements. Each node is equipped with two 2.3GHz hexacore Intel Xeon E5-2630 CPUs, 32GB of RAM, and 598GB HDD SCSI Dell (Raid) PERC H710 magnetic hard drives. Power measurements are collected every second from power meters[2]. The power meters are connected to a data collector via a serial link and provide measurements with an accuracy of 0.125 Watts. We are interested in identifying relationships between power consumption, task duration, CPU utilization, and volume of I/O. Detailed execution profiles (but without power/energy data) and performance analysis for both workflows can be found in [24].

Table 1 shows the execution profiles of Epigenomics and SoyKB tasks, with one row per task type. Since most Epigenomics tasks require 1 CPU core, power measurements were collected from a resource where only a single core was enabled (i.e., only 1 CPU slot is advertised by the resource manager). Only the `pileup` task requires 2 cores, but there is only one such task in the workflow. For SoyKB, many tasks require 2 CPU cores. Therefore, we collected power measurements from a resource configured with two cores. The last four columns in Table 1 show the average power consumption per task and the energy consumption to compute all tasks of that type in sequence (for both clusters). As power measurements were collected every second, measurements for tasks with very short runtimes (e.g., `sol2sanger` in the Epigenomics workflow) may not be accurate, and are thus not emphasized in our upcoming analyses. Note that both the Epigenomics and the SoyKB workflows are mostly composed of I/O-intensive tasks.

3. Workflow Energy Consumption Analysis

Energy-aware workflow scheduling studies [6, 7, 8, 9, 10, 11, 12, 13, 14, 15] typically assume that the power consumed by the execution of a task at time t, $P(t)$, is linearly related to the task's CPU utilization, $u(t)$, as:

$$P(t) = (P_{\max} - P_{\min}) \cdot u(t) \cdot \tfrac{1}{n}, \tag{1}$$

where $P_{\max}$ is the power consumption when the compute node is at its maximum utilization, $P_{\min}$ is the idle power consumption (i.e., when there is no or only background activity), and n is the number of cores on the compute node. ($(P_{\max} - P_{\min}) \cdot u(t)$ represents the dynamic power consumption.) Therefore, the energy consumption of the task, E, is defined as follows:

$$E = r \cdot P_{\min} + \int_0^r P(t)\mathrm{d}t, \tag{2}$$

where r denotes the task's runtime. To determine the idle power consumption $P_{\min}$, we collected power measurements on one node of the *taurus* and of the *orion* cluster at every second whenever no activity was performed on that node over a 2-month period (for a total of 216,000 measurements). The average idle power consumption from these measurements is 98.08W (standard deviation 1.77W) and 132.96W (standard deviation 2.47W), for *taurus* and *orion* respectively. Although

[2] Manufactured by OMEGAWATT: `http://www.omegawatt.fr/gb/index.php`

											Taurus			Orion		
Task	Count	#cores	Runtime		CPU util.		I/O Read		I/O Write		Power		Energy	Power		Energy
			μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	(Wh)	μ	σ	(Wh)
fastqSplit	7	1	5.8	1.9	99.8%	0.0	508.1	173.2	254.1	86.6	126.9	5.5	1.4	173.2	1.2	1.9
filterContams	140	1	1.2	0.2	99.1%	0.0	25.4	3.7	12.7	1.8	100.9	5.6	4.6	143.2	9.5	6.4
sol2sanger	140	1	0.4	0.1	95.7%	0.2	66.9	9.8	29.0	4.3	98.5	3.8	1.4	140.3	7.1	1.5
fast2bfq	140	1	0.8	0.1	97.8%	0.1	35.5	5.2	6.4	0.9	98.3	3.6	2.9	139.3	7.6	3.7
map	140	1	57.9	5.0	99.9%	0.0	437.9	2.4	2.6	0.6	126.8	0.9	285.7	168.4	0.7	379.6
mapMerge	8	1	5.9	6.9	99.5%	0.0	171.2	205.6	84.0	103.4	113.5	7.7	1.5	154.7	8.5	2.0
maqIndex	1	1	33.5	–	99.9%	–	511.7	–	338.3	–	125.1	–	1.2	167.7	–	1.6
pileup	1	2	38.4	–	80.8%	–	559.3	–	264.1	–	135.5	–	1.4	179.0	–	1.8

											Taurus			Orion		
Task	Count	#cores	Runtime		CPU util.		I/O Read		I/O Write		Power		Energy	Power		Energy
			μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	(Wh)	μ	σ	(Wh)
align_reference	25	2	1.8	0.0	53.9%	0.0	2609.7	0.0	186.6	0.01	134.8	4.7	1.6	178.9	3.2	2.1
sort_sam	25	2	1.3	0.1	61.9%	0.0	901.5	0.0	187.2	1.6	101.7	1.9	0.9	135.1	5.2	1.2
dedup	25	2	2.0	0.0	60.7%	0.0	901.9	0.0	186.9	0.2	106.2	4.3	1.5	149.3	5.9	2.0
add_replace	25	2	1.3	0.0	62.0%	0.0	901.5	0.0	186.9	0.0	102.6	1.7	0.9	134.3	4.8	1.2
realign_creator	25	2	133.1	2.6	75.9%	0.0	3230.8	8.7	189.6	2.8	135.3	0.3	125.1	179.4	0.6	166.0
indel_realign	25	1	34.3	0.0	18.9%	0.0	953.8	5.8	187.0	0.0	123.2	0.6	25.9	148.6	0.8	35.4
haplotype_caller	500	1	79.3	6.9	66.7%	0.0	1149.8	24.2	186.9	0.0	130.8	1.0	1329.5	163.2	1.2	1794.9
genotype_gvcfs	20	1	263.8	29.6	95.9%	0.0	1058.0	16.2	187.6	0.1	126.6	0.3	185.5	171.4	0.2	251.2
comb_variants	1	1	35.5	–	26.5%	–	958.0	–	186.9	–	108.9	–	1.1	153.8	–	1.5
variants_indel	1	2	48.6	–	23.7%	–	1699.5	–	454.4	–	114.0	–	1.5	157.0	–	2.1
filtering_indel	1	1	34.7	–	20.3%	–	955.2	–	186.9	–	109.1	–	1.0	151.8	–	1.5
variants_snp	1	2	48.6	–	23.2%	–	1699.5	–	454.4	–	115.4	–	1.5	155.0	–	2.1
filtering_snp	1	2	34.7	–	10.2%	–	955.3	–	186.9	–	109.6	–	1.0	152.3	–	1.5
merge_gcvf	1	1	46804.5	–	99.9%	–	3061.2	–	238.8	–	128.9	–	1675.3	171.9	–	2220.6

Table 1: Execution and energy profiles of the Epigenomics (top) and SoyKB (bottom) workflow tasks. Energy measurements are for running all tasks of that type in sequence. Runtimes are shown in seconds, I/O operations in MB, and power in W. (μ is the mean, and σ the standard deviation.)

		Epigenomics		SoyKB	
		Pearson	Spearman	Pearson	Spearman
CPU	*taurus*	0.38	0.55	-0.02	0.58
	orion	0.34	0.52	-0.08	0.58
I/O	*taurus*	0.86	0.44	0.64	0.40
	orion	0.79	0.38	0.68	0.49

Table 2: Pearson and Spearman coefficients for the correlation between power consumption and (i) CPU utilization and (ii) I/O volumes, for Epigenomics and SoyKB workflow tasks on the *taurus* and *orion* clusters.

both clusters have equivalent configurations, the measurements show difference in their idle power. In order to determine why the baseline performance differs between the two clusters, we contacted the system administrators of the Grid'5000 platform. The performance difference seems to be related to the NVIDIA Tesla M2075 GPU processor attached to each node of the Orion cluster, and thus should not impact power and energy consumption behaviors for the study conducted in this paper. This difference would be critical when analyzing hybrid CPU-GPU executions [25].

3.1. Correlation Analysis of Power Consumption

The power model in Eq. 1 does not consider the energy consumption of I/O operations, and hereafter we quantify the extent to which this omission makes the model inaccurate. Fig. 3 shows scatter plots of the power consumption versus CPU utilization for all task types of both workflows. Pearson and Spearman coefficient values for the correlation between power consumption and CPU utilization are shown in the top part of Table 2 for all task types for both workflows and both clusters. The coefficient values are low, meaning that no linear increase is observed in the power consumption as CPU utilization increases. Similarly, Spearman correlation values show moderate correlation, which indicates that power consumption is not monotonically related to CPU utilization. For example, the `align_reference` SoyKB task has an average CPU utilization at about 108% and consumes about 135W on *taurus*, while the `sort_sam` task from that same workflow has a CPU utilization at about 124% but consumes only 102W. This difference in power consumption is mostly explained by I/O operations (reads and writes). Fig. 4 shows scatter plots of the power consumption versus I/O read volumes per task and computational resource. The bottom part of Table 2 shows coefficient values for the correlation between power consumption and I/O volumes. In contrast to the CPU utilization analysis, coefficient values for the Pearson correlation analysis are higher, indicating that I/O volumes tend to be more linearly correlated to power consumption than CPU utilization for (most) tasks in our two workflows. Spearman's correlation coefficient values indicate weaker monotonic relationships.

These results show that power consumption is not strictly dependent, or even mainly influenced, by CPU utilization $u(t)$ (Eq. 1), but that it depends significantly on I/O volumes. Hence, we conduct a principal component analysis (PCA) to evaluate the variance of each parameter (CPU utilization, I/O reads, and I/O writes) and their impact on the power consumption. In this analysis, we aim to understand how CPU utilization and I/O operations are influencing (positively or negatively) power consumption, and quantify the weight of each parameter. From the principal components, i.e., sets of linearly uncorrelated variables, we obverse the *loadings* (the weight by which each standardized original variable should be multiplied to get the component score), which contain the data variance.

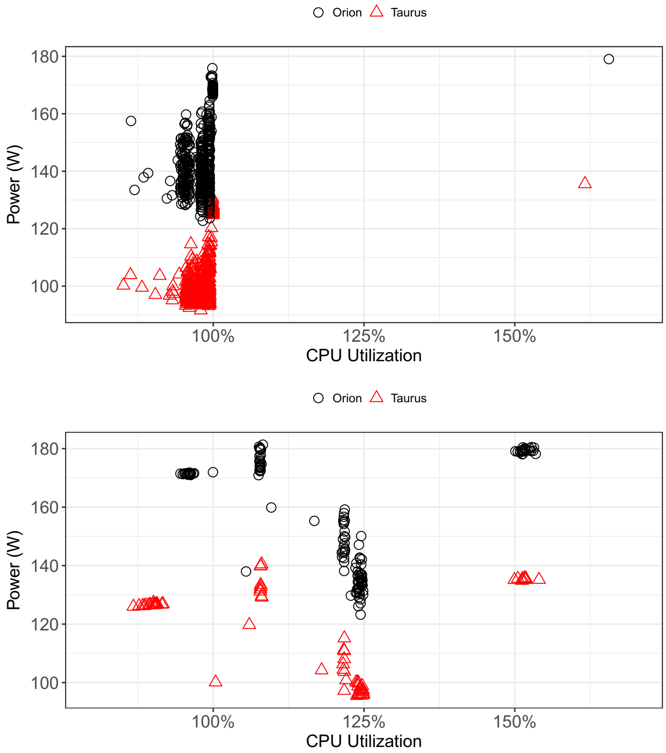

Figure 3: Task power consumption vs. CPU utilization for the Epigenomics (top) and SoyKB (bottom) workflows. Measurements for Epigenomics workflow are mostly for 1-core tasks, and 2-core tasks for SoyKB.

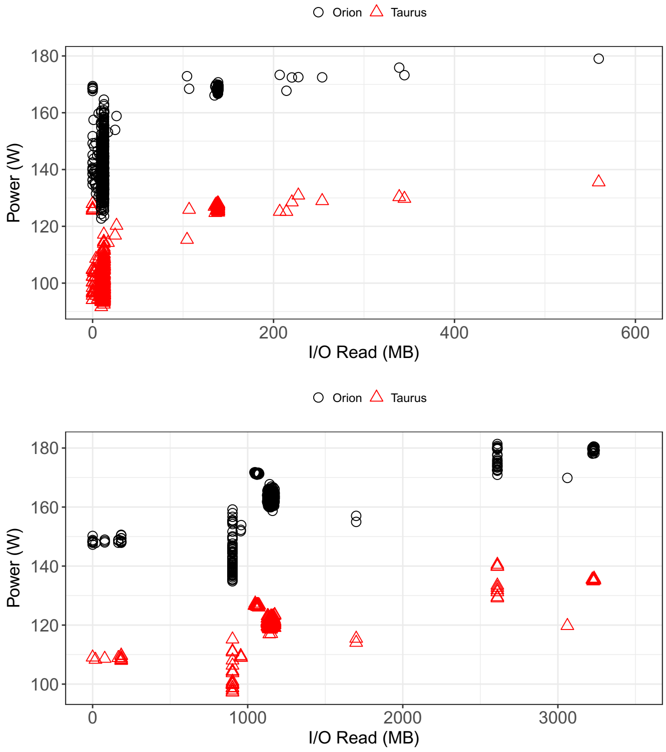

Figure 4: Task power consumption vs. I/O read for the Epigenomics (top) and SoyKB (bottom) workflows.

Parameter	Epigenomics			SoyKB		
	PC1	PC2	PC3	PC1	PC2	PC3
CPU Utilization	0.53	0.84	-0.03	-0.55	-0.62	0.56
I/O Reads	0.59	-0.35	0.71	-0.73	0.04	-0.67
I/O Writes	0.59	-0.40	-0.69	-0.39	0.78	0.47

Table 3: Principal component (PC) loadings (rotations) for the Epigenomics and SoyKB workflows.

Table 3 shows the principal component (PC) loadings (rotations) for each parameter. For Epigenomics, the first two PCs explain most of the variability (85.3%). All parameters present similar variance for PC1, with the I/O reads and I/O writes parameters dominating, while CPU utilization has greater impact on PC2. Since PC1 explains most of the variance (64.3%), the power consumption of the Epigenomics workflow is also significantly impacted by the number of I/O operations (in particular I/O reads) as shown in Fig. 4-top. Similarly, the first two PCs for SoyKB explain most of the variability (85.4%). I/O reads have greater impact on PC1, while PC2 is mostly impacted by CPU utilization and I/O writes. Although I/O reads have significant impact on PC1, this component only explains 49% of the variance, thus I/O reads have less influence on the power consumption for SoyKB (Fig. 4-bottom). Note that the impact of I/O reads on PC2 is minimal. Fig. 5 shows PCA biplots for both workflow parameters, which project the data on the first two PCs.

Overall, these results provide motivation and quantitative bases for developing a more accurate power model that captures the impact of I/O operations on power consumption in addition to that of CPU utilization.

4. Analysis of Power and Energy Consumption for Concurrent Task Execution

The power consumption model in Eq. 1 assumes that the consumed power is simply the CPU utilization divided by the number of cores. To evaluate the validity of this assumption we collected and analyzed power measurements for solitary and concurrent workflow task executions.

Since *taurus* and *orion* cluster nodes are all equipped with dual, hexacore CPUs, we performed task executions with two schemes for core allocation (see Fig. 6): (1) *unpaired*—cores are enabled in sequence on a single socket until all cores on that socket are enabled, and then cores on the next socket are enabled in sequence; and (2) *pairwise*—cores are enabled in round-robin fashion across sockets (i.e., each core is enabled on a different socket than the previously enabled core). We report on results for only a subset of workflow tasks because (1) some tasks are unique; (2) some task runtimes are very short and overheads in Pegasus, such as releasing the next task, make the benefit of running these tasks on multiple cores negligible; or (3) energy measurements may not be accurate for tasks with very short runtimes due to the measurements interval of 1s. Finally, all our results report average runtime, power and energy measurements for concurrent executions of instances of the same task type.

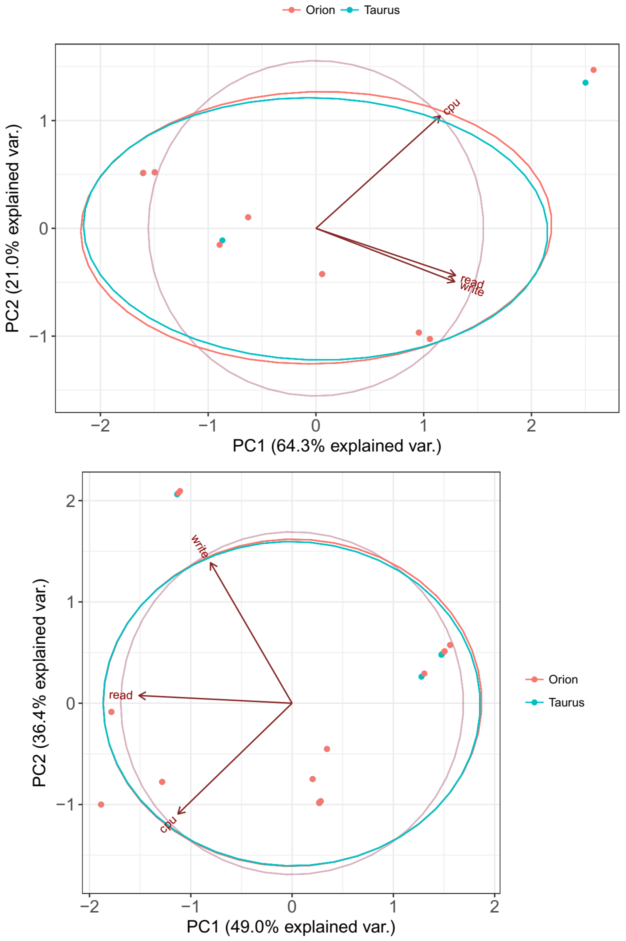

Figure 5: Principal component analysis biplot for the Epigenomics (top) and SoyKB (bottom) workflows.

4.1. *Epigenomics*

Fig. 7 shows the average task runtime, average task power consumption, and total energy consumption (i.e., to run all 140 tasks) when running `map` tasks concurrently for different numbers of CPU cores. Task performance is significantly impacted when multiple cores are used within a single socket. For example, when 2 cores are enabled in different sockets (*pairwise*), no performance decrease is observed. However, a performance degradation of about 25% occurs when both cores are within a single socket (*unpaired*). The above is due to the fact that each socket has a single L3 cache shared between its cores.

While the use of multiple cores within a single socket limits performance, it consumes less power per unit of time: on the order of 10% (Fig. 7-center). According to Eq. 1, power consumption should grow linearly. Instead, we observe that power consumption is not equally divided among the number of cores per CPU. Eq. 1 thus underestimates the energy usage per unit of time—averaged root mean squared error (RMSE) (for both clusters) is 10.57 for *pairwise* and 4.88 for *unpaired*.

The energy profile shown in Fig. 7-right accounts for the execution of all 140 `map` tasks. Although power consumption is lower when using a single socket, the total energy consumption is higher due to higher task runtimes. Workflow task executions may benefit from single socket CPU usage if task runtimes are very short. In this case, the performance loss is negligible and the difference of power consumption may save energy (e.g., the `filterContams` task in Epigenomics). The energy consumption for the set of `map` tasks presents a logarithmic decrease as a function of the number of cores. This logarithmic behavior is due to the increase in power consumption. The estimation errors propagated by Eq. 1 into Eq. 2 leads to averaged energy consumption estimation errors up to 24% (RMSEs are 0.02 for *pairwise* and 0.03 for *unpaired*).

4.2. *SoyKB*

Fig. 8 shows the average task runtime, the average task power consumption, and total energy consumption (i.e., to run all 500 tasks) when running `haplotype_caller` tasks concurrently using 2 up to 8 CPU cores. Due to disk space quota on Grid'5000, we were unable to run workflow instances that used more than 8 cores concurrently. We only report on results for more than 2 cores because the workflow cannot be executed on a single core. Task runtime differences between *unpaired* and *pairwise* is minimal regardless of the number of cores used. A small degradation in runtime is observed when the number of cores increase from 2 to 4. However, there is a significant performance decrease when the number of cores exceeds 4. This is because `haplotype_caller` performs substantial I/O operations (it only has 67% of CPU utilization on average). The performance degradation is due to simultaneous I/O operations, which cause tasks to idle due to I/O resources being unavailable and/or saturated. This idle time (IOWait) is reported in the logs generated by Pegasus.

Similar to Epigenomics, the *unpaired* scheme consumes slightly less power (about 5%, as seen in Fig. 8-center). The power consumption estimated by Eq. 1 lies between the real-world consumption with the two schemes, with averaged prediction errors up to 11% (RMSE up to 4.84 for *pairwise*). In Fig. 8-right, we see that the actual energy values are well above the estimated values (up to 23% higher for *orion*). The main factor for this discrepancy is I/O, including the time spent waiting for I/O operations to complete (as indicated by IOWait values in the Pegasus logs).

4.3. *Analysis of DVFS for Concurrent Task Execution*

Dynamic voltage and frequency scaling (DVFS) is a common optimization technique used to reduce energy consumption by exchanging runtime for lower energy usage. Several works [9, 10, 26] have adopted a cubic model to estimate power consumption as a function of the frequency [27], as follows:

$$P_C(t,f) = P_{f_{\min}} + (P_{\max} - P_{f_{\min}}) \cdot \left(\frac{f - f_{\min}}{f_{\min}}\right)^3, \qquad (3)$$

where f is the operating frequency in MHz, and $f_{\min}$ is the lowest operating frequency when hosts are idle. This model has been derived based on power measurements when running tasks at different frequencies, and used extensively in the literature. Fig. 9 shows the average power consumption for the `map` task

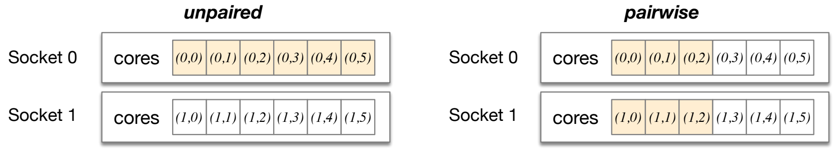

Figure 6: Example of CPU core usage for the *unpaired* (left) and *pairwise* (right) schemes when 6 cores are enabled.

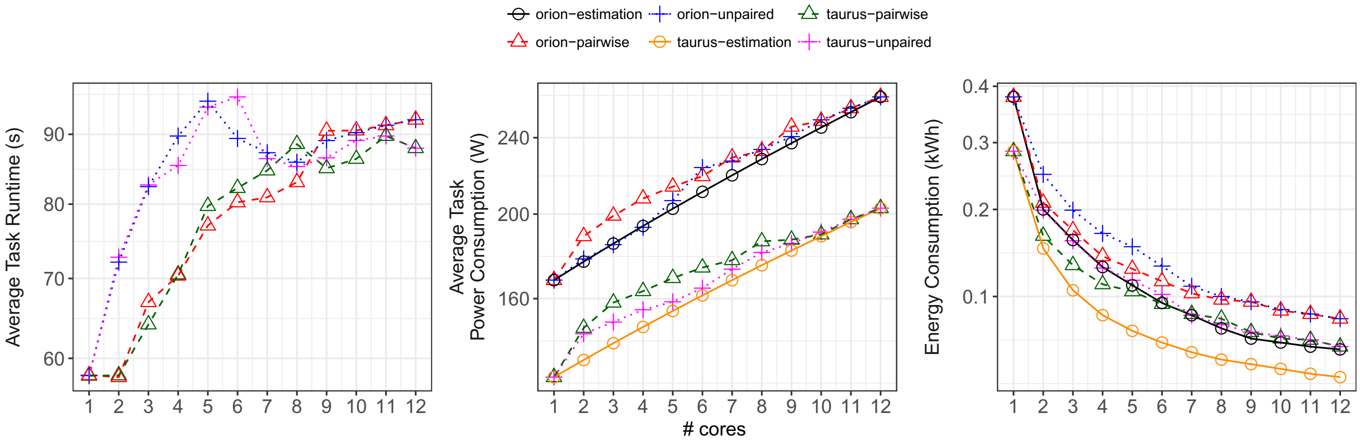

Figure 7: Average task runtime (left), average task power consumption (center), and energy consumption to run all 140 `map` tasks from Epigenomics (right). Power and energy consumption computed using Eqs. 1 and 2 are shown as `estimation`. Notice that although estimated power consumption when using mostly cores is nearly accurate, energy consumption is still imprecise due to variations on task runtimes.

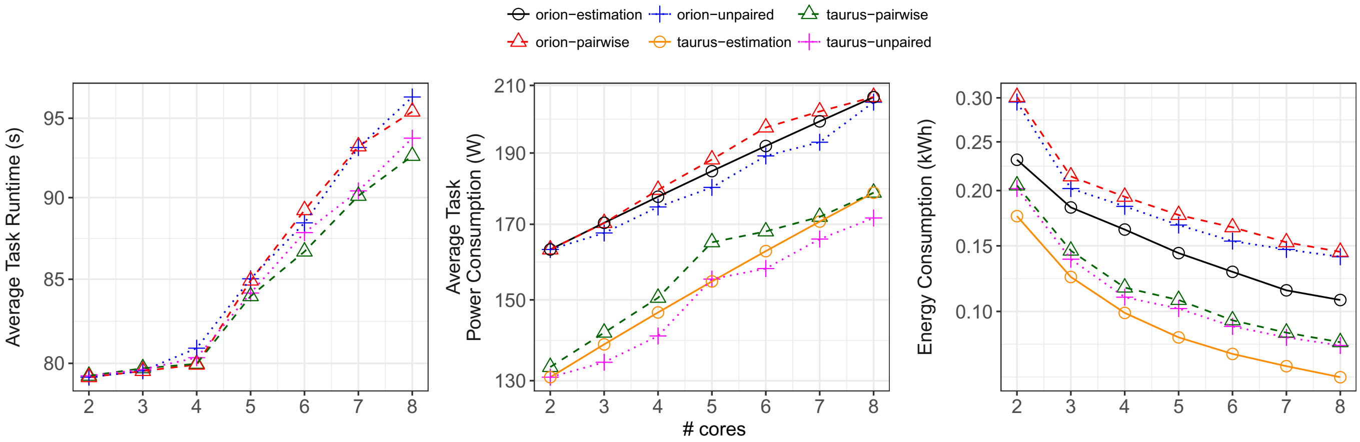

Figure 8: Average task runtime (left), average power consumption (center), and energy consumption to run all 500 `haplotype_caller` tasks from SoyKB (right). Power and energy consumption computed using Eqs. 1 and 2 are shown as `estimation`.

for different number of CPU cores and operating frequencies when executed on the *taurus* cluster (frequencies range from 1200MHz to 2300MHz). As expected, power consumption increases with frequency and the number of cores. We also observe that trends are similar regardless of whether cores are allocated in *pairwise* or *unpaired* fashion. A similar study was also performed for the `haplotype_caller` task for SoyKB (omitted here due to space constraints), which yielded similar outcomes. The cubic model can be fitted to all these experimental results, so as to model power consumption for given number of cores and operating frequencies. We find that, regardless of the proportion of I/O operations performed and/or IOWait imposed to the task execution, the cubic model provides high accuracy (i.e., RMSE $\ll 1$ for the higher frequencies, and RMSE $\ll 2$ for the lower frequencies).

We conclude that the power consumption of effects of DVFS on workflow task executions are orthogonal to the I/O and core allocation effects discussed in previous sections. Therefore, when deriving our model in the next section, we do not consider DVFS explicitly, but merely use the operating frequency as an input to the model.

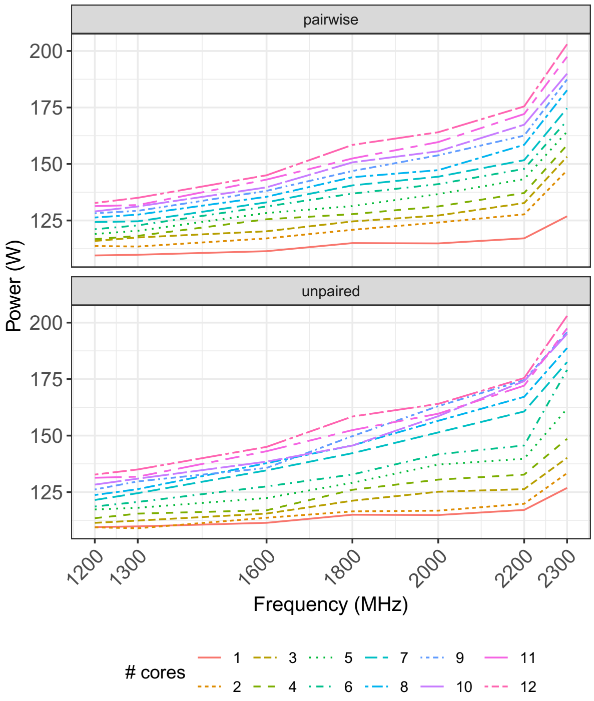

Figure 9: Average power consumption vs. operating frequency for `map` tasks of Epigenomics when running with different number of cores activated.

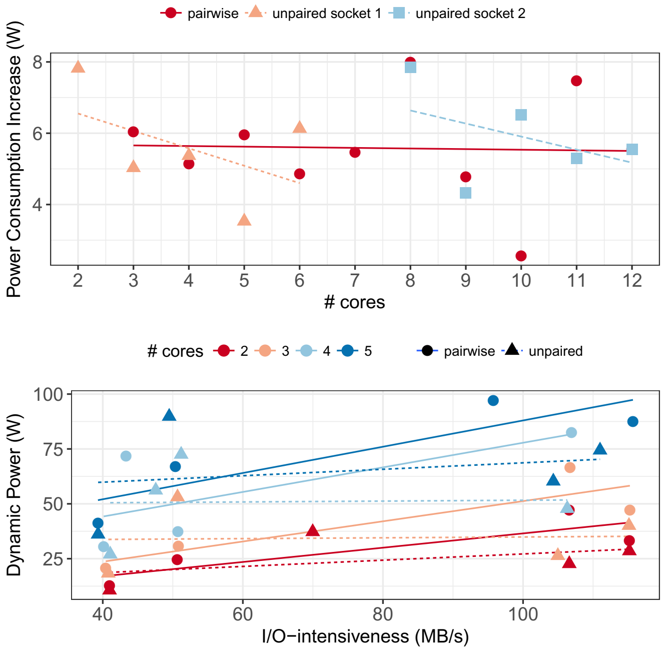

Figure 10: Linear regression models. *Top:* power consumption increase in function of number of cores enabled for the Epigenomics `map` tasks. *Bottom:* dynamic power consumption vs. I/O-intensiveness for the SoyKB `realign_creator`, `indel_realign`, `haplotype_caller`, and `genotype_gvcfs` tasks.

5. Modeling and Simulating Energy Consumption of I/O-intensive Workflows

In this section, we present a model for power consumption that accounts for I/O in addition to CPU utilization. This model also accounts for the number of cores and the way in which they are activated (*unpaired* or *pairwise* schemes), as well as for the time spent waiting for I/O operations to complete.

5.1. Model

We model $P(t)$, the power consumption of a compute node at time t, as:

$$P(t) = P_{\text{CPU}}(t) + P_{\text{I/O}}(t), \tag{4}$$

where $P_{\text{CPU}}(t)$, resp. $P_{\text{I/O}}(t)$, is the power consumption due to CPU utilization, resp. I/O operations. In what follows, we detail the model for both these terms.

CPU – Let s denote the number of sockets on the compute node, and n the number of cores per socket, so that the total number of cores on the compute node is $s \cdot n$. Let K denote the set of tasks that use at least one core on the compute node. We have:

$$P_{\text{CPU}}(t) = \sum_{k,i,j} P_{\text{CPU}}(k, i, j, t), \tag{5}$$

where $P_{\text{CPU}}(k, i, j, t)$ is the power consumption of CPU utilization at time t due to the execution of task k ($k \in K$) on socket i ($0 \leq i < s$) at core j ($0 \leq j < n$) on the compute node.

In previous sections, we examined the impact of cores/socket allocation on power consumption in addition to CPU utilization. We have seen that the power consumption does not have constant increase as cores are enabled on sockets, and the behavior depends on the scheme used to enable further cores (*pairwise* or *unpaired*). Fig. 10-*top* shows a scatter plot of power consumption increase for each additional enabled core for the `map` task of the Epigenomics workflow. The increase for the *unpaired* scheme can be approximated by linear regression with negative slope. For the *pairwise* scheme, an approximation by linear regression leads to nearly constant increase (noting that the RMSE is relatively high). Note that each data point in Fig. 10-*top* is in fact 140 almost perfectly superposed data points for the 140 `map` task executions (due to nearly negligible standard deviations, see Table 1). Although this figure is for a particular task of the Epigenomics workflow, very similar results are obtained for all tasks for both production workflows considered in this work. Therefore, we derive a model that is only dependent on the task's CPU utilization and the hardware platform.

Based on the above, we model $P_{\text{CPU}}(k, i, j, t)$ as:

$$P_{\text{CPU}}(k,i,j,t) = \begin{cases} (P_{\max} - P_{\min}) \cdot \frac{u(t)}{s \cdot n} & \text{if } j = 0 \text{ (first core on a socket)} \\ 0.881 \cdot P_{\text{CPU}}(k,i,j-1,t) & \text{if } j > 0 \text{ and } \textit{pairwise} \\ 0.900 \cdot P_{\text{CPU}}(k,i,j-1,t) & \text{if } j > 0 \text{ and } \textit{unpaired} \end{cases} \tag{6}$$

where $u(t)$ is the task's CPU utilization at time t (which can be computed by benchmarking the task on a dedicated compute node). The model is written recursively as the power consumption due to enabling a core on a socket depends on the power consumption due to previously enabled cores on that socket. The 0.881 and 0.900 constants above are obtained from the aforementioned linear regressions. Finally, note that

$P_{\text{CPU}}(k, i, j, t)$ does not depend on i since only the rank (j) of a core in a socket matters.

I/O – Similarly to the above model for power consumption due to CPU utilization, we have:

$$P_{\text{I/O}}(t) = \sum_{k,i,j} P_{\text{I/O}}(k, i, j, t), \tag{7}$$

where $P_{\text{I/O}}(k, i, j, t)$ is the power consumption of I/O operations at time t due to the execution of task k ($k \in K$) on socket i ($0 \leq i < s$) at core j ($0 \leq j < n$) on the compute node.

Fig. 10-*bottom* shows dynamic power consumption (i.e., power consumption beyond P_{min}) vs. I/O-intensiveness for 4 tasks of the SoyKB workflow (`realign_creator`, `indel_realign`, `haplotype_caller`, and `genotype_gvcfs`). We define the I/O-intensiveness as the I/O volume (for reads and writes) in MB divided by the time the task spends performing solely computation (i.e., the runtime minus the time for performing and waiting for I/O operations). A higher value indicates a more I/O-intensive task, as it represents I/O overhead per second of CPU usage. We are able to compute the I/O-intensiveness of each task based on profiling data in Pegasus logs. The four task types in Fig. 10 exhibit a range of CPU utilizations, with relatively high volumes of data read/written. As for the results in Fig. 10, similar results are obtained for all tasks in the workflows we consider. We use a linear regression, shown in the figure, which has positive slope regardless of the core allocation scheme (with a steeper slope for the *pairwise* scheme). Based on these results, we model $P_{\text{I/O}}(k, i, j, t)$ as follows:

$$P_{\text{I/O}}(k,i,j,t) = \begin{cases} 0.486 \cdot (1+0.317 \cdot \omega(t)) \cdot P_{\text{CPU}}(k,i,j,t) & \text{if } \textit{pairwise} \\ 0.213 \cdot (1+0.317 \cdot \omega(t)) \cdot P_{\text{CPU}}(k,i,j,t) & \text{otherwise} \end{cases} \tag{8}$$

where the 0.486 and 0.213 values above come from the linear regressions, and $\omega(t)$ is 0 if I/O resources are not saturated at time t, or 1 if they are (i.e., idle time due to IOWait). More precisely, $\omega(t)$ is equal to 1 whenever the volume of I/O requests placed by concurrently running tasks exceeds some platform-dependent maximum I/O throughput. When using this model, e.g., to drive simulations of workflow task executions so as to evaluate energy-efficient workflow scheduling algorithms, it is then necessary to keep track of simulated I/O requests so as to set the $\omega(t)$ value accordingly. It turns out that, in our results, the impact of IOWait does not show any strong correlation with the features of different task types. This is why $\omega(t)$ in Eq. 8 is weighted by a single factor (0.317). We computed this factor as the average of the most accurate such factor values we computed individually for each task type. Our evaluation of the model (see Section 5.2) shows that it achieves high accuracy across task types. It is thus tempting to claim that the impact of the IOWait effect on power consumption can be captured reasonably well using a single, application-independent value for the above factor. Providing a definitive answer as to whether this claim is general would require repeating these experiments using other workflow applications.

5.2. Experimental Evaluation

To evaluate the accuracy of our model, we extended a simulator [28] of the state-of-the-art Pegasus [18] workflow management system (WMS), which is the WMS we used to perform the experiments described in Section 2. This simulator is built using the WRENCH framework [29], a framework for implementing simulators of WMSs that are accurate and can run scalably on a single computer, while requiring minimal software development effort [30]. We extended the simulator by replacing its simulation model for power consumption (the traditional model in Eq. 1) by the model proposed in Section 5.1. We provide the simulator with a description of the hardware specifications of the *taurus* and *orion* Grid'5000 clusters and with traces from individual Epigenomics and SoyKB workflow task executions. As a result, our simulator can simulate the exact procedure used for obtaining all real-world experimental results described in previous sections, making it possible to draw direct comparisons between real-world and simulated results. The simulator code, details on the simulation calibration procedure, and experimental scenarios used in the rest of this section are all publicly available online [28].

Fig. 11 shows the simulated power and energy consumption measurements as well as with the traditional model based on Eqs. 1 and 2 (shown as `estimation`) and with our proposed model (shown as `wrench-*`) for the *taurus* cluster. Due to space constraints, we only show results for the `map` Epigenomics task, and the `haplotype_caller` and `indel_realign` SoyKB tasks. For the `map` tasks, the RMSE for *pairwise* is 4.24, and 3.49 for *unpaired*, which improves the accuracy over the traditional model by about two orders of magnitude for the former and one half for the latter. Similarly, RMSEs for the `haplotype_caller` tasks are 2.86 and 2.07 for *pairwise* and *unpaired* respectively, or improvements of about two orders of magnitude for both schemes. Last, RMSEs for the `indel_realign` tasks are 0.59 for *pairwise* and 0.47 for *unpaired*, or improvements by about an order of magnitude. Predicted energy consumption based on our proposed model nearly matches the actual measurements for both schemes for all task types (RMSEs $\ll 0.01$).

Fig. 12 shows the simulated power and energy consumption measurements for the *orion* cluster. For the `map` tasks, the RMSE for *pairwise* is 2.87, and 1.12 for *unpaired*, which improves over the traditional model by up to two orders of magnitude. RMSEs for the `haplotype_caller` tasks are 1.92 and 1.63 for *pairwise* and *unpaired* respectively, or improvements of up to three orders of magnitude. Finally, RMSEs for the `indel_realign` tasks are 0.53 for *pairwise* and 0.21 for *unpaired*, or improvements by about two orders of magnitude. Similarly to the *taurus* cluster, predicted energy consumption based on our proposed model nearly match the actual measurements for both schemes for all task types (RMSEs $\ll 0.01$).

In addition to evaluating the accuracy of our model w.r.t. the actual measurements, we also assess its precision when compared with a state-of-the-art power and energy consumption model for cloud platforms, DEM [31]. We have chosen this model due to its usage by the scientific community for developing resource provisioning algorithms in cloud systems [32, 33];

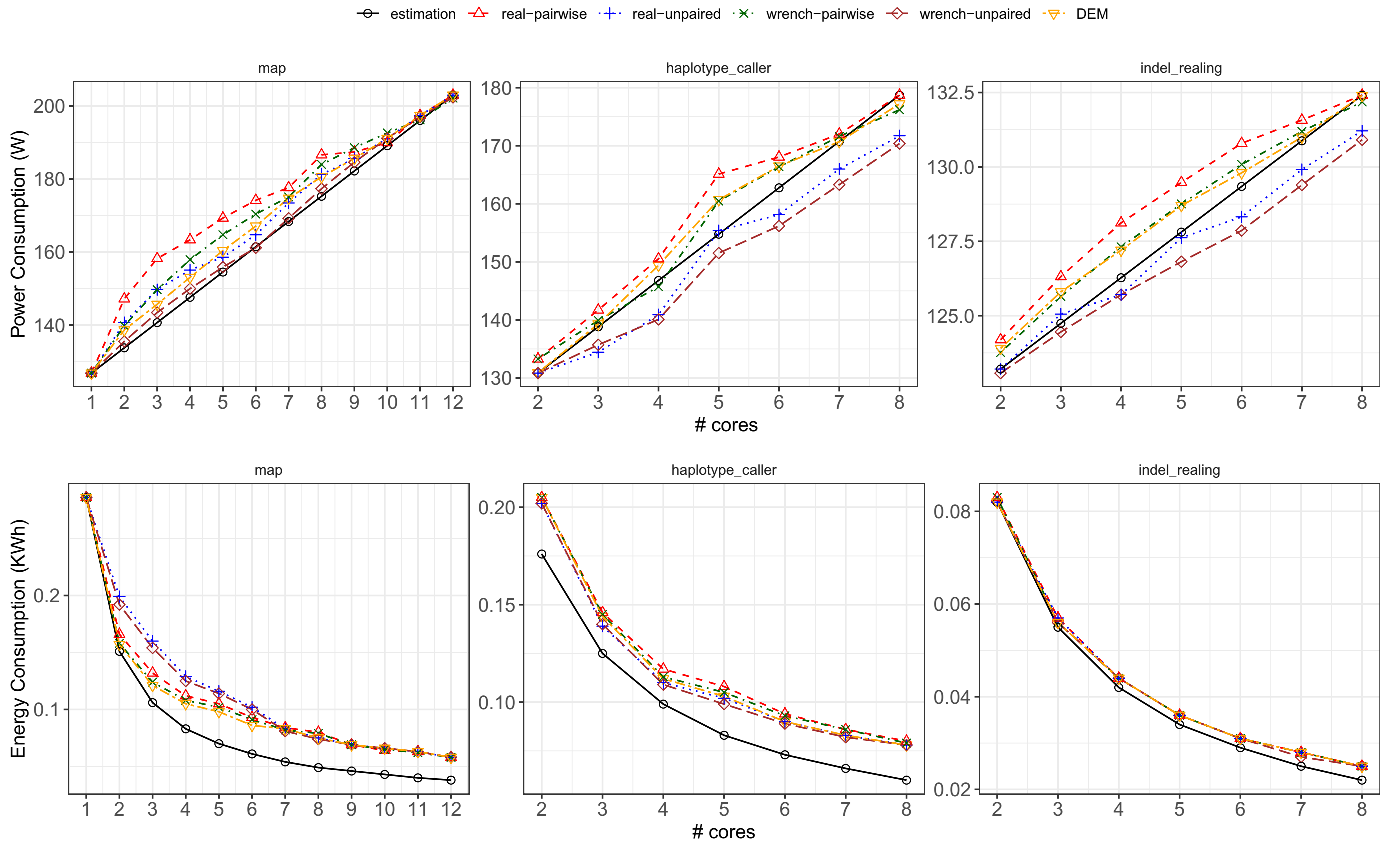

Figure 11: *Taurus*: Per-task power (top) and total energy (bottom) consumption measurements for the Epigenomics `map` task and the SoyKB `haplotype_caller` and `indel_realign`, as well as estimated with Eq. 1 and 2 (`estimation`), our proposed model (`wrench-*`), and a state-of-art model from the literature (`DEM`).

and the reduced number of required system-level parameters for instantiating the model (see Section 7 for a discussion of limitations of other I/O-aware models). DEM proposes a disk power model based on multivariate thresholds and distinguishing I/O modes for standard magnetic hard drives. We have executed simple benchmarks that use sequential and random I/O modes on both clusters (*taurus* and *orion*) to determine an averaged number of disk operations per second. We have then used DEM for predicting the power and energy usage for the workflows tasks shown in Figs. 11 and 12. Power and energy consumptions estimated by DEM are reasonably close to those for `wrench-pairwise`, but in most cases the latter are closer to actual measurements. This suggests that the DEM model was derived based on experiments that allocate cores using the *pairwise* scheme. Indeed, DEM substantially overestimates power consumption when cores are allocated using the *unpaired* scheme (RMSE $\gg$ 10). Because our model accounts for the core allocation scheme it produced good results for either scheme.

The overall conclusion is that out model affords considerable improvement over the current power and energy consumption models widely used in the workflow scheduling literature. As these models do not account for core allocation schemes nor for I/O, they either overestimate or underestimate power and energy consumptions. Estimating the energy consumption of the entire workflow execution is useful (Figs. 11-*bottom* and 12-*bottom*). But per-task power consumption (Figs. 11-*top* and 12-*top*) is more relevant for energy-aware workflow scheduling and application-level resource provisioning strategies implemented by workflow management systems, as decisions are made at the task level. Our proposed model, due to its higher accuracy, affords a better basis than the standard model used in the workflow scheduling literature to make these decisions. This said, we note that our model still tends to always underestimate (often marginally) task power consumptions.

6. Energy-Aware Opportunities and Challenges for Workflow Optimizations

Most previously proposed energy-aware workflow optimizations achieve some desirable tradeoff between workflow makespan and energy consumption. Typically, the goal is to obtain non-trivial energy savings by allowing for small increases in execution times. Common techniques include DVFS and workload consolidation. Although application-level approaches yield reasonable savings, the gain can be small when compared to infrastructure-level strategies to reduce energy usage. A trivial such infrastructure-level strategy consists in picking the most energy-efficient platform among multiple platforms that offer comparable application performance capabil-

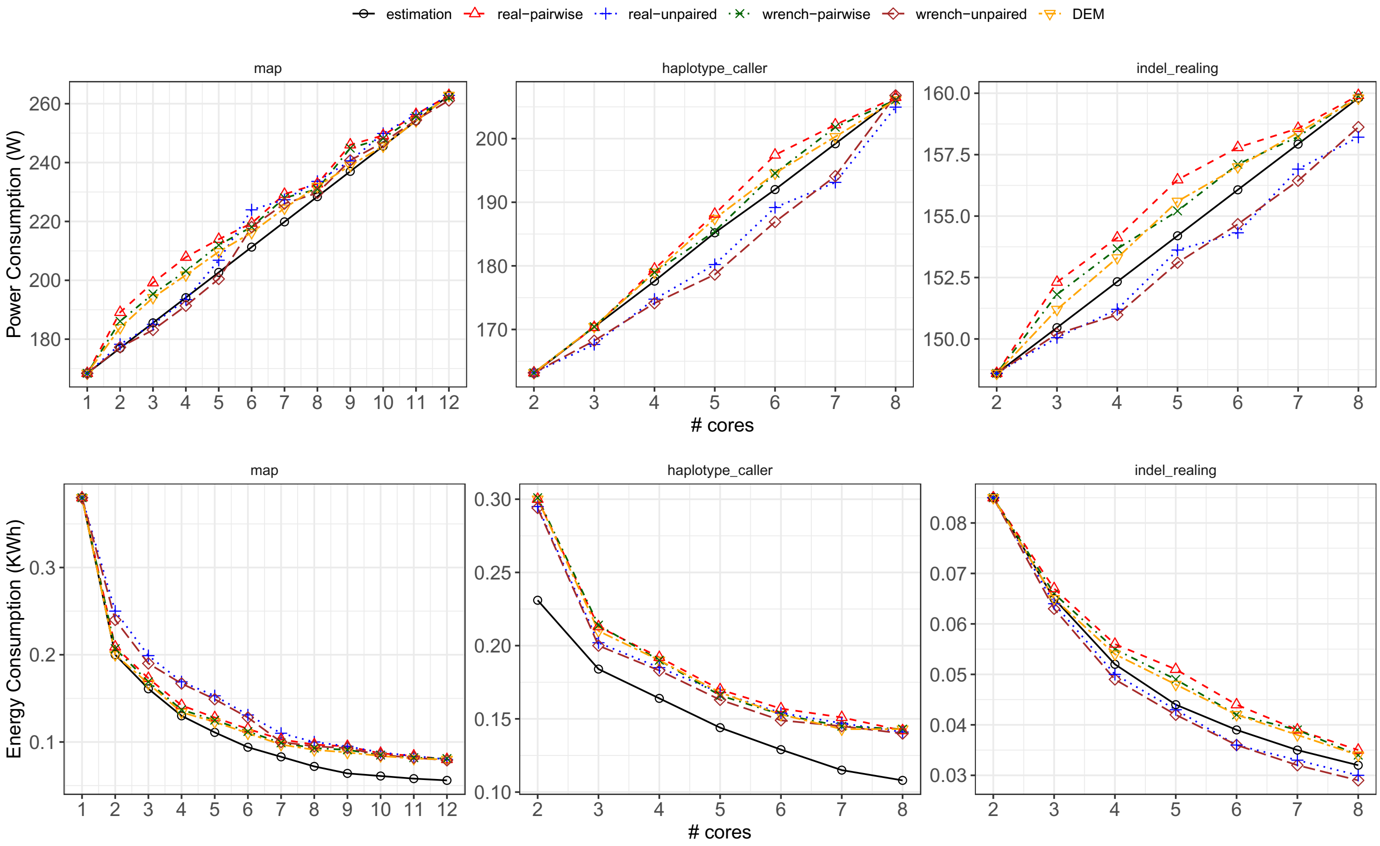

Figure 12: *Orion*: Per-task power (top) and total energy (bottom) consumption measurements for the Epigenomics `map` task and the SoyKB `haplotype_caller` and `indel_realign`, as well as estimated with Eq. 1 and 2 (`estimation`), our proposed model (`wrench-*`), and a state-of-art model from the literature (`DEM`).

ities. It turns out that this strategy is applicable to the platforms considered in this work. While both the *orion* and *taurus* clusters leads to almost identical performance for our production workflows (see the performance profiles in [24]), *taurus* executions can consume up to 20% less energy when compared to *orion* executions. This is because the GPUs installed on the compute nodes of *orion* have a higher idle power consumption than that on *taurus* (see Section 3). Since our workflow tasks do not use GPUs, this higher idle power consumption causes a net loss (i.e., higher energy consumption and no performance benefit).

Experimental results show that the amount of I/O operations has a significant impact on energy consumption. However, all measurements in this paper were obtained from compute nodes equipped with standard, magnetic hard drives. A study evaluating the energy usage of data-intensive workflows [34], shows that the use of solid state drives, NVRAM, or DRAM may significantly reduce the amount of energy consumed. Therefore, I/O operations may have less influence on energy usage for platforms where such hardware is available. But ARM processors are also known to be more energy-efficient than x86 processors. For platforms that use such architectures I/O operations may again become a major factor in power consumption. Overall, we claim that in many relevant cases, given the complex and constantly changing hardware environments, power and energy usage models must take into account both I/O and CPU usage (as seen in the previous section).

The energy reduction impact obtained by infrastructure-level optimizations imposes challenges for optimizations at the application-level as follows:

- *Infrastructure-level optimizations have broader impact* – The manufacturing of energy efficient devices has gained significant attention recently. Large data-centers have incorporated these technologies to optimize energy usage across a wide array of applications and workloads. Workflow optimizations, in comparison, are often limited to a specific use-case, workload, or workflow management system. As such, application-level optimizations need to have big payoffs in order to justify spending time and resources on them instead of simply relying on infrastructure-level optimizations. Furthermore, application optimizations, no matter how carefully designed, may quickly become obsolete due to rapidly evolving hardware infrastructures.

- *Application-level optimizations require realistic models* – Energy efficient workflow executions depend primarily on how resources are allocated and how tasks are scheduled. However, current algorithms for computing resource allocations and task schedules are impeded by the lack of accurate information about the resource requirements of workflows [24, 35]. Many resource requirement estimation techniques are based on analytical models or statistical techniques. Ideally, these models would fit most workflows and be able to capture their performance behavior accurately. But simplifying assumptions are often necessary and results in significant losses of accuracy. Consequently, there is a need for modeling methods that can build accurate models automatically. This work is a first effort towards this approach, where the measurements collected from the experiments allowed the development of fine-grained workflow profiles, which in turn lead to more realistic power and energy models that are validated empirically. Although this method at first glance appears to be more complex than the previous model (that requires only $P_{\max}$ and $P_{\min}$, see Equation 1), the use of automated calibration tools may lead to a simpler process, while providing more accurate results. Although we have not conducted a comprehensive study to quantify the number of measurements needed for defining the constant factors in Equations 6 and 8, the results in this paper are obtained with a number of core-hours needed to derive these constants that is negligible when compared to the computational demands of current and emerging workflow applications – recall that our proposed model is platform-dependent, but application-agnostic.

7. Related Work

In the past few years, green computing has become a major topic of discussion in the scientific computing community. Many recent studies have addressed green solutions, in particular on distributed computing platforms. Research efforts in this field commonly include powering off or putting idle machines into low power states based on predictions of future workloads. On the application side, efforts are mainly focused on the optimization of resource provisioning and workload scheduling constrained by budgets and application deadlines.

A recent survey [5] of techniques for improving energy-efficiency describes methods to evaluate and model the energy consumed by resources on distributed systems. The survey presents taxonomies of compute node and network energy-aware techniques classified according to the technology employed. These techniques include adjustment of the processor's frequency and power consumption through DVFS [9], workload consolidation by running multiple tasks on the same physical machine in order to reduce the number of nodes that are powered on [36], energy-aware task scheduling [37, 38], virtual machine migration [4, 39], the coordination of network protocols [40], etc. These strategies often model energy consumption merely as a function of runtime and/or do not consider the performance loss due to running multiple tasks within a socket.

Several models have been developed to predict the power consumption of distributed system workloads. Most of them focus on measuring the resource utilization of distributed systems [41, 42, 43]. In [44], an integrated power consumption model, which incorporates previous approaches into a single model, describes a distributed system where several clients issue requests to a central storage server. Most of these models are limited to single-core and energy consumption is related to CPU usage. More recently, some models consider data transfers and/or I/O operations during task execution [45, 46, 47, 31]. However, these models require low-level system specific parameters (e.g., operating frequency, voltage noise, runtime estimations without I/O, sequential and random I/O speeds, etc.) that are typically not available at the workflow management system level (which only relies on basic profiling at the application level). Our proposed model, instead, leverages performance metrics that are currently captured by production workflow management systems at runtime, and thus can be used directly by these system to make sound energy-aware workflow scheduling decisions.

In the context of scientific workflows, several works [6, 7, 8, 9, 10, 11, 12, 13, 14, 15] have proposed energy-aware algorithms for task scheduling or resource provisioning. These algorithms are often designed to meet energy budget or deadline constraints. Their model assumes that the total energy usage is equal to the integral of the consumed power, which is linearly related to the resource utilization. In this work, we have shown that I/O operations also have significant impact on the power consumption, and thereby the energy. To the best of our knowledge, this is the first work that profiles and analyzes power and energy consumption of real scientific workflow applications at a fine-grained level, and proposes a model that accounts not only for CPU usage but also for cores/sockets allocation schemes and for I/O usage.

8. Conclusion and Future Work

In this work, we have profiled and analyzed the power consumption of two production scientific workflow applications executed on distributed platforms. We have investigated the impact of CPU utilization and I/O operations on energy usage, as well as the impact of executing multiple tasks concurrently on multi-socket, multi-core compute nodes. In contrast to traditional power consumption models used in the energy-efficient workflow scheduling literature, we find that power consumption is impacted non-linearly by the way in which cores in sockets are allocated to workflow tasks. Furthermore, our experimental results show that I/O operations have significant impact on power consumption. Based on these results, we proposed a power model for I/O intensive workflows that accounts for the above phenomena. Experimental evaluation of this model shows that it accurately captures real-world behavior, with up to two orders of magnitude improvement over the traditional model widely used in the workflow scheduling literature.

In future work, we plan to instantiate and validate our proposed model for other workflows and platform configurations. In particular, we hope to use power-metered platforms in which compute nodes use storage technology other than HDDs, e.g., SSDs, NVRAM, burst buffers. With these technologies, the impact of I/O on power consumption may exhibit different behaviors that could mandate augmenting our model. The power consumption of I/O could also be smaller relative to that of computation, but platforms that target extreme-scale computing tend to employ low-power compute nodes (i.e., equipped with ARM processors). More generally, it would be interesting to see how our model can be instantiated for scenarios in which workflow tasks use different hardware components on a compute node (e.g., hardware accelerators). Another future work goal is to extend the synthetic workflow generator in [48], which produces realistic synthetic workflow configurations based on profiles extracted from workflow execution traces. The objective is to extend the generated workflow descriptions to include data obtained from real-world power profiles, so that these descriptions are sufficient for instantiating the power consumption model proposed in this work. Finally, a natural future direction is to apply the results in this work to energy-aware application scheduling and resource provisioning strategies, and to quantify the resulting benefits.

Acknowledgments

This work is funded by NSF contracts #1642369 and #1642335, “SI2-SSE: WRENCH: A Simulation Workbench for Scientific Workflow Users, Developers, and Researchers”, and NSF contracts #1923539 and #1923621, “CyberTraining: Integrating core CI literacy and skills into university curricula via simulation-driven activities”, and CNRS under grant #PICS07239; and partly funded by NSF contract #1664162, and DOE contract #DE-SC0012636. Experiments presented in this paper were carried out using the Grid’5000 testbed, supported by a scientific interest group hosted by Inria and including CNRS, RENATER and several Universities as well as other organizations (see `https://www.grid5000.fr`).

References

[1] I. J. Taylor, E. Deelman, D. B. Gannon, M. Shields, et al., Workflows for e-Science: scientific workflows for grids, Vol. 1, Springer, 2007.

[2] M. Romanus, P. K. Mantha, M. McKenzie, T. C. Bishop, E. Gallichio, A. Merzky, Y. El Khamra, S. Jha, The anatomy of successful ecss projects: Lessons of supporting high-throughput high-performance ensembles on xsede, in: Proceedings of the 1st Conference of the Extreme Science and Engineering Discovery Environment: Bridging from the eXtreme to the campus and beyond, ACM, 2012, p. 46.

[3] J. Baliga, R. W. Ayre, K. Hinton, R. S. Tucker, Green cloud computing: Balancing energy in processing, storage, and transport, Proceedings of the IEEE 99 (1) (2010) 149–167.

[4] A. Beloglazov, R. Buyya, Y. C. Lee, A. Zomaya, A taxonomy and survey of energy-efficient data centers and cloud computing systems, in: Advances in computers, Vol. 82, Elsevier, 2011, pp. 47–111.

[5] A.-C. Orgerie, M. D. d. Assuncao, L. Lefevre, A survey on techniques for improving the energy efficiency of large-scale distributed systems, ACM Computing Surveys (CSUR) 46 (4) (2014) 47.

[6] I. Pietri, M. Malawski, G. Juve, E. Deelman, J. Nabrzyski, R. Sakellariou, Energy-constrained provisioning for scientific workflow ensembles, in: 2013 International Conference on Cloud and Green Computing, IEEE, 2013, pp. 34–41.

[7] J. J. Durillo, V. Nae, R. Prodan, Multi-objective workflow scheduling: An analysis of the energy efficiency and makespan tradeoff, in: 2013 13th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing, IEEE, 2013, pp. 203–210.

[8] T. Thanavanich, P. Uthayopas, Efficient energy aware task scheduling for parallel workflow tasks on hybrids cloud environment, in: International Computer Science and Engineering Conference (ICSEC), 2013, pp. 37–42.

[9] T. Guérout, T. Monteil, G. Da Costa, R. N. Calheiros, R. Buyya, M. Alexandru, Energy-aware simulation with dvfs, Simulation Modelling Practice and Theory 39 (2013) 76–91.

[10] I. Pietri, R. Sakellariou, Energy-aware workflow scheduling using frequency scaling, in: International Conference on Parallel Processing Workshops (ICCPW), 2014.

[11] E. N. Watanabe, P. P. Campos, K. R. Braghetto, D. M. Batista, Energy saving algorithms for workflow scheduling in cloud computing, in: 2014 Brazilian Symposium on Computer Networks and Distributed Systems, IEEE, 2014, pp. 9–16.

[12] D. Shepherd, I. Pietri, R. Sakellariou, Workflow scheduling on power constrained vms, in: Proceedings of the 8th International Conference on Utility and Cloud Computing, IEEE Press, 2015, pp. 420–421.

[13] Z. Li, J. Ge, H. Hu, W. Song, H. Hu, B. Luo, Cost and energy aware scheduling algorithm for scientific workflows with deadline constraint in clouds, IEEE Transactions on Services Computing 11 (4) (2015) 713–726.

[14] M. Ghose, P. Verma, S. Karmakar, A. Sahu, Energy efficient scheduling of scientific workflows in cloud environment, in: 2017 IEEE 19th International Conference on High Performance Computing and Communications; IEEE 15th International Conference on Smart City; IEEE 3rd International Conference on Data Science and Systems (HPCC/SmartCity/DSS), IEEE, 2017, pp. 170–177.

[15] T. Wu, H. Gu, J. Zhou, T. Wei, X. Liu, M. Chen, Soft error-aware energy-efficient task scheduling for workflow applications in dvfs-enabled cloud, Journal of Systems Architecture 84 (2018) 12–27.

[16] R. Ferreira da Silva, A.-C. Orgerie, H. Casanova, R. Tanaka, E. Deelman, F. Suter, Accurately simulating energy consumption of i/o-intensive scientific workflows, in: Computational Science – ICCS 2019, Springer International Publishing, 2019, pp. 138–152. `doi:10.1007/978-3-030-22734-0_11`.

[17] D. Balouek, A. C. Amarie, G. Charrier, F. Desprez, E. Jeannot, E. Jeanvoine, A. Lèbre, D. Margery, N. Niclausse, L. Nussbaum, et al., Adding virtualization capabilities to the grid5000 testbed, in: International Conference on Cloud Computing and Services Science, Springer, 2012, pp. 3–20.

[18] E. Deelman, K. Vahi, G. Juve, M. Rynge, S. Callaghan, P. J. Maechling, R. Mayani, W. Chen, R. Ferreira da Silva, M. Livny, K. Wenger, Pegasus, a workflow management system for science automation, Future Generation Computer Systems 46 (2015) 17–35. `doi:10.1016/j.future.2014.10.008`.

[19] M. Albrecht, P. Donnelly, P. Bui, D. Thain, Makeflow: A portable abstraction for data intensive computing on clusters, clouds, and grids, in: Proceedings of the 1st ACM SIGMOD Workshop on Scalable Workflow Execution Engines and Technologies, ACM, 2012, p. 1.

[20] T. Fahringer, A. Jugravu, S. Pllana, R. Prodan, C. Seragiotto Jr, H.-L. Truong, Askalon: a tool set for cluster and grid computing, Concurrency and Computation: Practice and Experience 17 (2-4) (2005) 143–169.

[21] G. Juve, A. Chervenak, E. Deelman, S. Bharathi, G. Mehta, K. Vahi, Characterizing and profiling scientific workflows, Future Generation Computer Systems 29 (3) (2013) 682–692. `doi:10.1016/j.future.2012.08.015`.

[22] T. Joshi, B. Valliyodan, S. M. Khan, Y. Liu, J. M. dos Santos, Y. Jiao, D. Xu, H. T. Nguyen, N. Hopkins, M. Rynge, et al., Next generation resequencing of soybean germplasm for trait discovery on xsede using pegasus workflows and iplant infrastructure (2014).

[23] G. Juve, B. Tovar, R. Ferreira da Silva, D. Król, D. Thain, E. Deelman, W. Allcock, M. Livny, Practical resource monitoring for robust high throughput computing, in: 2nd Workshop on Monitoring and Analysis for High Performance Computing Systems Plus Applications (HPCMASPA’15), 2015, pp. 650–657. `doi:10.1109/CLUSTER.2015.115`.

[24] R. Ferreira da Silva, G. Juve, M. Rynge, E. Deelman, M. Livny, Online task resource consumption prediction for scientific workflows, Parallel Processing Letters 25 (3) (2015) 1541003. `doi:10.1142/S0129626415410030`.

[25] Y. Inadomi, T. Patki, K. Inoue, M. Aoyagi, B. Rountree, M. Schulz, D. Lowenthal, Y. Wada, K. Fukazawa, M. Ueda, et al., Analyzing and mitigating the impact of manufacturing variability in power-constrained supercomputing, in: SC’15: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, IEEE, 2015, pp. 1–12. `doi:10.1145/2807591.2807638`.

[26] F. Cao, M. M. Zhu, Energy-aware workflow job scheduling for green clouds, in: IEEE International Conference onGreen Computing and Communications (GreenCom), 2013.

[27] J.-M. Pierson, H. Casanova, On the utility of dvfs for power-aware job placement in clusters, in: Euro-Par 2011 Parallel Processing, Springer, 2011.

[28] WRENCH Pegasus Simulator, https://github.com/wrench-project/pegasus (2019).

[29] The WRENCH Project, http://wrench-project.org/ (2019).

[30] H. Casanova, S. Pandey, J. Oeth, R. Tanaka, F. Suter, R. Ferreira da Silva, WRENCH: A Framework for Simulating Workflow Management Systems, in: 13th Workshop on Workflows in Support of Large-Scale Science (WORKS'18), 2018, pp. 74–85. doi:10.1109/WORKS.2018.00013.

[31] W. Lin, H. Wang, Y. Zhang, D. Qi, J. Z. Wang, V. Chang, A cloud server energy consumption measurement system for heterogeneous cloud environments, Information Sciences 468 (2018) 47–62. doi:10.1016/j.ins.2018.08.032.

[32] M. Kumar, S. Sharma, A. Goel, S. Singh, A comprehensive survey for scheduling techniques in cloud computing, Journal of Network and Computer Applicationsdoi:10.1016/j.jnca.2019.06.006.

[33] H. A. Kholidy, An intelligent swarm based prediction approach for predicting cloud computing user resource needs, Computer Communicationsdoi:10.1016/j.comcom.2019.12.028.

[34] G. Haldeman, I. Rodero, M. Parashar, S. Ramos, E. Z. Zhang, U. Kremer, Exploring energy-performance-quality tradeoffs for scientific workflows with in-situ data analyses, Computer Science-Research and Development 30 (2) (2015) 207218. doi:10.1007/s00450-014-0268-6.

[35] R. Ferreira da Silva, G. Juve, E. Deelman, T. Glatard, F. Desprez, D. Thain, B. Tovar, M. Livny, Toward fine-grained online task characteristics estimation in scientific workflows, in: 8th Workshop on Workflows in Support of Large-Scale Science, WORKS '13, 2013, pp. 58–67. doi:10.1145/2534248.2534254.

[36] L. Lefevre, A.-C. Orgerie, Towards energy aware reservation infrastructure for large-scale experimental distributed systems, Parallel Processing Letters 19 (03) (2009) 419–433. doi:10.1142/S0129626409000316.

[37] L. Wang, S. U. Khan, D. Chen, J. KołOdziej, R. Ranjan, C.-Z. Xu, A. Zomaya, Energy-aware parallel task scheduling in a cluster, Future Generation Computer Systems 29 (7) (2013) 1661–1670. doi:10.1016/j.future.2013.02.010.

[38] D. Kliazovich, P. Bouvry, S. U. Khan, Dens: data center energy-efficient network-aware scheduling, Cluster computing 16 (1) (2013) 65–75. doi:10.1007/s10586-011-0177-4.

[39] H. Liu, C.-Z. Xu, H. Jin, J. Gong, X. Liao, Performance and energy modeling for live migration of virtual machines, in: Proceedings of the 20th international symposium on High performance distributed computing, ACM, 2011, pp. 171–182. doi:10.1007/s10586-011-0194-3.

[40] S. He, J. Chen, D. K. Yau, H. Shao, Y. Sun, Energy-efficient capture of stochastic events under periodic network coverage and coordinated sleep, IEEE Transactions on Parallel and Distributed Systems 23 (6) (2011) 1090–1102. doi:10.1109/TPDS.2011.242.

[41] Y. C. Lee, A. Y. Zomaya, Energy efficient utilization of resources in cloud computing systems, The Journal of Supercomputing 60 (2) (2012) 268–280. doi:10.1007/s11227-010-0421-3.

[42] T. Enokido, M. Takizawa, An extended power consumption model for distributed applications, in: IEEE International Conference on Advanced Information Networking and Applications (AINA), 2012, pp. 912–919. doi:10.1109/AINA.2012.90.

[43] T. Samak, C. Morin, D. Bailey, Energy consumption models and predictions for large-scale systems, in: 2013 IEEE International Symposium on Parallel & Distributed Processing, Workshops and Phd Forum, IEEE, 2013, pp. 899–906. doi:10.1109/IPDPSW.2013.228.

[44] T. Enokido, M. Takizawa, An integrated power consumption model for distributed systems, IEEE Transactions on Industrial Electronics 60 (2) (2013) 824–836. doi:10.1109/TIE.2012.2206357.

[45] N. P. Jouppi, A. B. Kahng, N. Muralimanohar, V. Srinivas, Cacti-io: Cacti with off-chip power-area-timing models, IEEE Transactions on Very Large Scale Integration (VLSI) Systems 23 (7) (2014) 1254–1267. doi:10.1109/TVLSI.2014.2334635.

[46] M. Dorier, O. Yildiz, S. Ibrahim, A.-C. Orgerie, G. Antoniu, On the energy footprint of i/o management in exascale hpc systems, Future Generation Computer Systems 62 (2016) 17–28. doi:10.1016/j.future.2016.03.002.

[47] K. Tang, X. He, S. Gupta, S. S. Vazhkudai, D. Tiwari, Exploring the optimal platform configuration for power-constrained hpc workflows, in: 2018 27th International Conference on Computer Communication and Networks (ICCCN), IEEE, 2018, pp. 1–9. doi:10.1109/ICCCN.2018.8487322.

[48] R. Ferreira da Silva, W. Chen, G. Juve, K. Vahi, E. Deelman, Community resources for enabling and evaluating research in distributed scientific workflows, in: 10th IEEE International Conference on e-Science (eScience'14), 2014, pp. 177–184. doi:10.1109/eScience.2014.44.