Accurately Simulating Energy Consumption of I/O-intensive Scientific Workflows

Rafael Ferreira da Silva[1], Anne-Cécile Orgerie[2], Henri Casanova[3], Ryan Tanaka[3], Ewa Deelman[1], and Frédéric Suter[4]

[1] USC Information Sciences Institute, Marina del Rey, CA, USA
[2] Univ Rennes, Inria, CNRS, IRISA, Rennes, France
[3] Information and Computer Sciences, University of Hawaii, Honolulu, HI, USA
[4] IN2P3 Computing Center, CNRS, Villeurbanne, France
{rafsilva,deelman}@isi.edu, anne-cecile.orgerie@irisa.fr,
{ryanyt,henric}@hawaii.edu, frederic.suter@cc.in2p3.fr

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 have 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 often comprise individual computational (but often I/O-intensive) tasks with some dependency structure, which is why many scientists today formulate their computational problems as scientific workflows [26]. To obtain simulation results within acceptable time-frames, large scientific workloads are executed on distributed computing infrastructures such as grids and clouds [21]. The need to manage energy consumption across the entire suite of information and communication technology has received significant attention in the last few years [1, 3]. 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 [19]. In the context of scientific workflows, researchers have proposed a range of energy-aware workflow task scheduling or resource provisioning algorithms [8, 17, 20, 23, 30]. 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. 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 distributed platform that comprises multi-socket, multi-core compute nodes equipped with power meters. We select two widely scientific workflows, each of which has some I/O-intensive tasks, for which we conduct a comprehensive analysis of the power and energy consumption of their execution. Via this analysis, we quantify the accuracy, or lack thereof, of power consumption models commonly used in the energy-efficient workflow scheduling literature. We then propose a more accurate power consumption model. More specifically, this work makes the following contributions:

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 workflow scheduling research;
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;
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.

2 Workflow Characterization

The analysis presented in this work is based on the execution of two production scientific workflow applications on the Grid'5000 [2] 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. We consider these two I/O-intensive workflows:

- *Epigenomics* [12]: 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. We consider an Epigenomics instance with 577 tasks.
- *SoyKB* [11]: A bioinformatics workflow that re-sequences soybean germplasm lines selected for desirable traits such as oil, protein, soybean cyst nematode resistance, stress resistance, and root system architecture. We consider a SoyKB instance with 676 tasks.

We profiled these workflows when executed with Pegasus [5], a state-of-the-art workflow management system. Pegasus monitors and logs fine-grained profiling data such as I/O operations, runtime, memory usage, and CPU utilization [13].

The workflows were executed on the *taurus* cluster at the Grid'5000 Lyon site, which is instrumented at the node level with power meters. We used a single node to run the workflow tasks and collect power measurements (although not efficient, it allowed us to collect non-biased measurements). Each node is equipped with two 2.3GHz hexacore Intel Xeon E5-2630 CPUs, 32GB of RAM, and standard magnetic hard drives. Power measurements are collected every second from power meters (with an accuracy of 0.125 Watts[5]) that are connected to a data collector via a serial link. 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 [25].

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 two columns in Table 1 show the average power consumption per task and the energy consumption to compute all tasks of that type in sequence. As power measurement were collected every second, tasks with very short runtimes (e.g., `sol2sanger` in the Epigenomics workflow) may not allow accurate power measurements, and are not emphasized in our upcoming analyses.

3 Workflow Energy Consumption Analysis

Energy-aware workflow scheduling studies [8,17,20,23,30] 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:

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

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

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

Task	Count	#cores	Runtime		CPU util.		I/O Read		I/O Write		Power		Energy (Wh)
			μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	
fastqSplit	7	1	5.8	1.9	99.8%	0.0	508.1	173.2	254.1	86.6	126.9	5.48	1.4
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
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
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
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
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
maqIndex	1	1	33.5	–	99.9%	–	511.7	–	338.3	–	125.1	–	1.2
pileup	1	2	38.4	–	80.8%	–	559.3	–	264.1	–	135.5	–	1.4

Task	Count	#cores	Runtime		CPU util.		I/O Read		I/O Write		Power		Energy (Wh)
			μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	
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
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
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
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
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
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
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
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
comb_variants	1	1	35.5	–	26.5%	–	958.0	–	186.9	–	108.9	–	1.1
variants_indel	1	2	48.6	–	23.7%	–	1699.5	–	454.4	–	114.0	–	1.5
filtering_indel	1	1	34.7	–	20.3%	–	955.2	–	186.9	–	109.1	–	1.0
variants_snp	1	2	48.6	–	23.2%	–	1699.5	–	454.4	–	115.4	–	1.5
filtering_snp	1	2	34.7	–	10.2%	–	955.3	–	186.9	–	109.6	–	1.0
merge_gcvf	1	1	46804.5	–	99.9%	–	3061.2	–	238.8	–	128.9	–	1675.3

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.)

only background activity), and n is the number of cores on the compute node. 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 our 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).

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. 1 shows scatter plots of the power consumption versus CPU utilization for all task types of both workflows. We observe very low Pearson's correlation coefficient values between power consumption and CPU utilization (0.38 for Epigenomics, -0.02 for SoyKB). This means that no linear increase is observed in the power consumption as CPU utilization increases. For example, the `align_reference` SoyKB task has an average CPU utilization at about 108% and consumes about 135W, 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 volumes of I/O (reads and writes). Fig. 2 shows scatter plots of the power consumption versus I/O read volumes per task and computational resource. In contrast to the CPU utilization

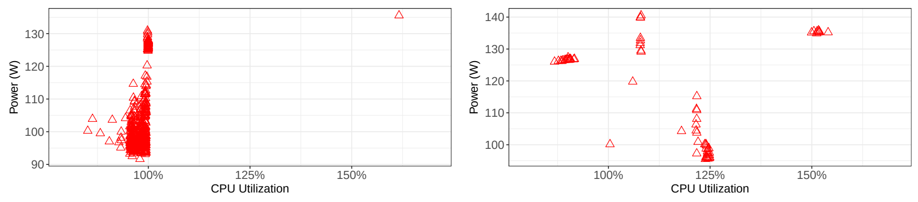

Fig. 1. Task power consumption vs. CPU utilization for the Epigenomics (left) and SoyKB (right) workflows.

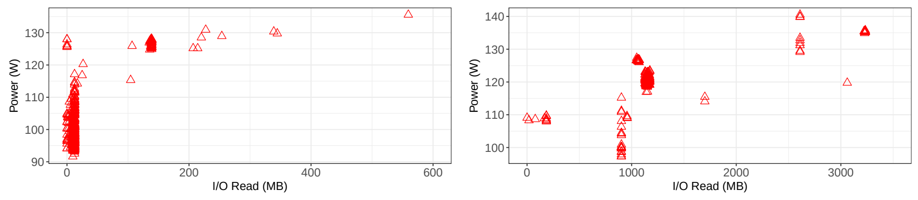

Fig. 2. Task power consumption vs. I/O read for the Epigenomics (left) and SoyKB (right) workflows.

analysis, Pearson's correlation coefficient values are 0.86 for Epigenomics, and 0.64 for SoyKB.

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 consequently quantify the weight of each parameter. From the principal components, set of values 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.

Table 2 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 read and I/O write 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. 2-left. Similarly, the first two PCs for SoyKB explain most of the variability (85.4%). I/O read has greater impact on PC1, while PC2 is mostly impacted by CPU utilization and I/O write. Although I/O read has significant impact on PC1, this component only explains 49% of the variance, thus I/O read has less influence on the power consumption for SoyKB (Fig. 2-right). Note that the impact of I/O read on PC2 is minimal.

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

Table 2. Principal component (PC) loadings (rotations) for the Epigenomics and SoyKB workflows.

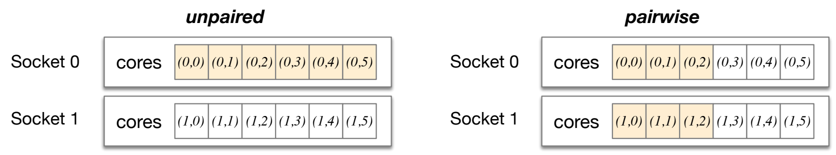

Fig. 3. Example of CPU core usage for the *unpaired* (left) and *parwise* (right) schemes when 6 cores are enabled.

Overall, these results provide motivation and quantitative bases for developing a more accurate power model that captures the implications 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 our cluster nodes are all equipped with dual, hexacore CPUs, we performed task executions with two schemes for core allocation (see Fig. 3): (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 in parallel 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.

Epigenomics: Fig. 4 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.

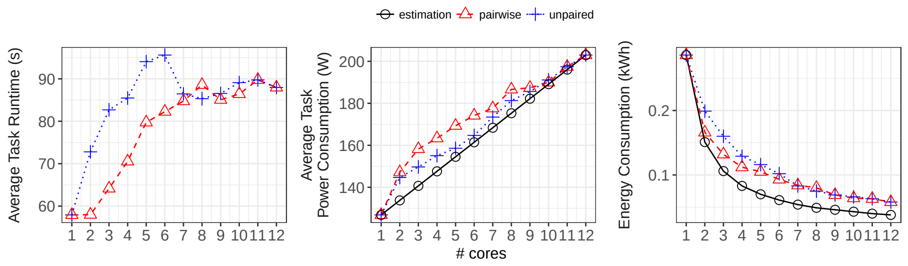

Fig. 4. Average task runtime (left), average task power consumption (center), and energy consumption to run all 140 `map` tasks from Epigenomics (right). Power and energy consumption computed using Eqs. 1 and 2 are shown as `estimation`.

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. 4-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—root mean squared error (RMSE) for *pairwise* is 10.64, and 4.92 for *unpaired*.

The energy profile shown in Fig. 4-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 energy consumption estimation errors up to 23% (RMSEs are 0.02 for *pairwise* and 0.03 for *unpaired*).

SoyKB: Fig. 5 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 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.

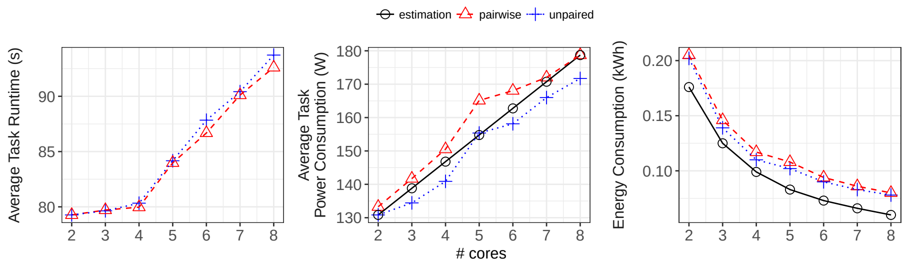

Fig. 5. Average task runtime (left), average power consumption (center), and energy consumption to run all 500 `haplotype_caller` tasks from SoyKB (right). Power and energy consumption computed using Eqs. 1 and 2 are shown as `estimation`.

Similar to Epigenomics, the *unpaired* scheme consumes slightly less power (about 5%, as see in Fig. 5-center. The power consumption estimated by Eq. 1 lies between the real-world consumption with the two schemes, with prediction errors up to 10% (RMSE up to 4.85 for *pairwise*). In Fig. 5-right, we see that the actual energy values are well above the estimated values (up to 22% higher). The main factor for this discrepancy is I/O operations, including the time spent waiting for I/O to complete (as indicated by IOWait values in the Pegasus logs).

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

In this section, we present an augmented 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 (IOWait).

5.1 Model

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

$$P(t) = P_{\mathrm{CPU}}(t) + P_{\mathrm{I/O}}(t), \tag{3}$$

where $P_{\mathrm{CPU}}(t)$, resp. $P_{\mathrm{I/O}}(t)$, 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 the set of tasks that use at least one core on the compute node. We have:

$$P_{\mathrm{CPU}}(t) = \textstyle\sum_{k,i,j} P_{\mathrm{CPU}}(k,i,j,t), \tag{4}$$

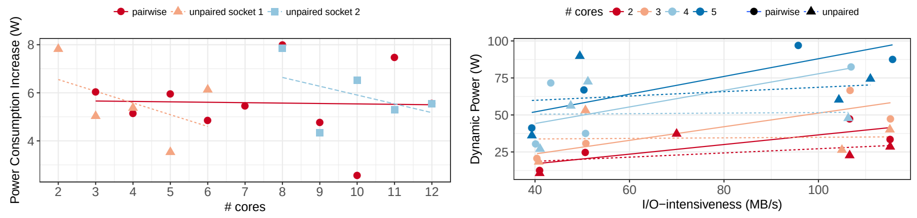

Fig. 6. Linear regression models. *Left:* power consumption increase in function of number of cores enabled for the Epigenomics `map` tasks. *Right:* dynamic power consumption vs. I/O-intensiveness for the SoyKB `realign_creator`, `indel_realign`, `haplotype_caller`, and `genotype_gvcfs` tasks.

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 ($0 \leq j < n$).

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. 6-*left* 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). 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 now define our model for $P_{\text{CPU}}(k,i,j,t)$ as:

$$P_{\text{CPU}}(k,i,j,t) = \begin{cases} (P_{\max} - P_{\min}) \cdot \dfrac{u(t)}{s \cdot n} & \text{if } j = 0 \text{ (first core on a socket)} \\ 0.881 \cdot P_{\text{CPU}}(k,i,j-1,t) & \text{if } j > 0 \text{ and } \textit{pairwise} \\ 0.900 \cdot P_{\text{CPU}}(k,i,j-1,t) & \text{if } j > 0 \text{ and } \textit{unpaired} \end{cases} \tag{5}$$

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:

$$P_{\text{I/O}}(t) = \textstyle\sum_{k,i,j} P_{\text{I/O}}(k,i,j,t), \tag{6}$$

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. 6-*right* 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. 6 exhibit a range of CPU utilizations, with relatively high volumes of data read/written. As for the results in Fig. 6, 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:

$$P_{\text{I/O}}(k,i,j,t)=\begin{cases} 0.486\cdot(1+0.317\cdot\omega(t))\cdot P_{\text{CPU}}(k,i,j,t) & \text{if } \textit{pairwise} \\ 0.213\cdot(1+0.317\cdot\omega(t))\cdot P_{\text{CPU}}(k,i,j,t) & \text{otherwise} \end{cases} \tag{7}$$

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). $\omega(t)$ 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 result, the impact of IOWait does not show any strong correlation with the features of different task types. This is why $\omega(t)$ in Eq. 7 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 a comprehensive set of experiments with more workflow applications running under this condition.

5.2 Experimental Evaluation

To evaluate the accuracy of our model, we extended a simulator [29] of the state-of-the-art Pegasus [5] 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 simulator framework [28], which can be used to

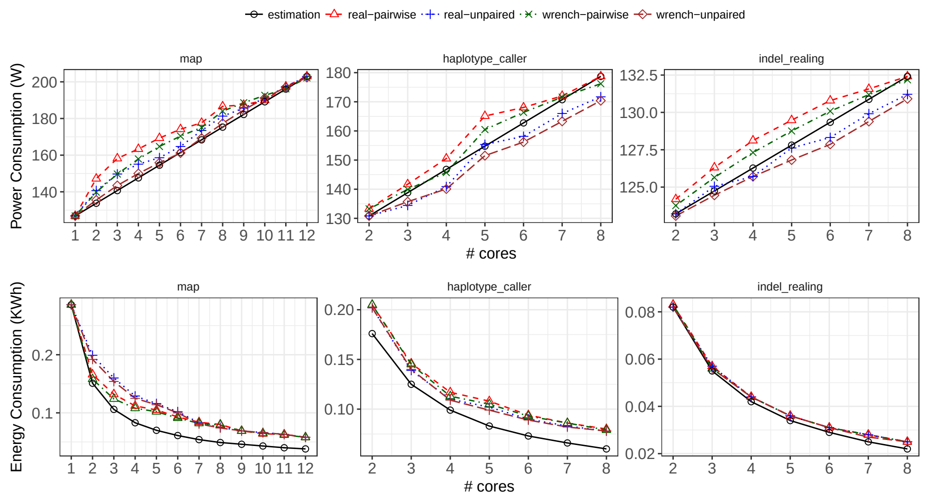

Fig. 7. Per-task power (top) and total energy (bottom) consumption measurements for the Epigenomics `map` task and the SoyKB `haplotype_caller` and `indel_realign`, as well as estimated with Eq. 1 and 2 (`estimation`) and our proposed model (`wrench-*`).

build simulators of WMSs that are accurate, can run scalably on a single computer, and can be implemented with minimal software development effort [4]. 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* Grid'5000 cluster 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 [29].

Fig. 7 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-*`). 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 over the traditional model by about two orders of magnitude for the former, and a half for the later. 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 match the actual measurements for both schemes for all task types (RMSEs $\ll 0.01$).

6 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 [19] 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 [16], energy-aware task scheduling [14, 27], virtual machine migration [3, 18], the coordination of network protocols [10], etc. These strategies often model energy consumption as a function of runtime, or do not consider the performance loss of 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 [6, 15, 22]. In [7], 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. A few models consider data transfers, but as a separate operation (I/O operations during task execution are not considered).

In the context of scientific workflows, several works [8, 17, 20, 23, 30] 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 also accounts for cores/sockets allocation and I/O usage.

7 Conclusion and Future Work

In this work, we have profiled and analyzed the power consumption of two production scientific workflow applications executed on a distributed platform. We have investigated the impact of resource utilization and I/O operations on the 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 model 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 behaviors. Experimental evaluation of this model shows that it accurately captures real-world behavior, with order of magnitude improvement over the traditional model.

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 have SSDs instead of HDDs. With SSDs, 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 note that platforms that target extreme-scale computing also often employ low-power compute nodes (i.e., equipped with ARM processors). Another future work goal is to extend the synthetic workflow generator in [24], 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 that is sufficient to instantiate the power consumption model proposed in this work.

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 CNRS under grant #PICS07239; and partly funded by NSF contract #1664162, and DOE contract #DE-SC0012636.

References

1. Baliga, J., et al.: Green cloud computing: Balancing energy in processing, storage, and transport. Proceedings of the IEEE **99**(1), 149–167 (2011)
2. Balouek, D., et al.: Adding virtualization capabilities to the grid5000 testbed. In: International Conference on Cloud Computing and Services Science (2012)
3. Beloglazov, A., et al.: A taxonomy and survey of energy-efficient data centers and cloud computing systems. Advances in computers **82**(2) (2011)
4. Casanova, H., et al.: WRENCH: A Framework for Simulating Workflow Management Systems. In: 13th Workshop on Workflows in Support of Large-Scale Science (WORKS'18) (2018)
5. Deelman, E., et al.: Pegasus, a workflow management system for science automation. Future Generation Computer Systems **46**, 17–35 (2015)
6. Enokido, T., Takizawa, M.: An extended power consumption model for distributed applications. In: IEEE International Conference on Advanced Information Networking and Applications (AINA). pp. 912–919 (2012)
7. Enokido, T., Takizawa, M.: An integrated power consumption model for distributed systems. IEEE Transactions on Industrial Electronics **60**(2), 824–836 (2013)
8. Ghose, M., et al.: Energy efficient scheduling of scientific workflows in cloud environment. In: IEEE International Conference on High Performance Computing and Communications (HPCC). pp. 170–177 (2017)

9. Guérout, T., et al.: Energy-aware simulation with dvfs. Simulation Modelling Practice and Theory **39** (2013)
10. He, S., et al.: Energy-efficient capture of stochastic events under periodic network coverage and coordinated sleep. IEEE Transactions on Parallel and Distributed Systems **23**(6), 1090–1102 (2012)
11. Joshi, T., et al.: Next Generation Resequencing of Soybean Germplasm for Trait Discovery on XSEDE using Pegasus Workflows and iPlant Infrastructure. In: XSEDE (2014), poster
12. Juve, G., et al.: Characterizing and profiling scientific workflows. Future Generation Computer Systems **29**(3), 682–692 (2013)
13. Juve, G., et al.: Practical resource monitoring for robust high throughput computing. In: Workshop on Monitoring and Analysis for High Performance Computing Systems Plus Applications (HPCMASPA'15). pp. 650–657 (2015)
14. Kliazovich, D., Bouvry, P., Khan, S.U.: Dens: data center energy-efficient network-aware scheduling. Cluster computing **16**(1) (2013)
15. Lee, Y.C., Zomaya, A.Y.: Energy efficient utilization of resources in cloud computing systems. The Journal of Supercomputing **60**(2), 268–280 (2012)
16. Lefevre, L., Orgerie, A.C.: Towards energy aware reservation infrastructure for large-scale experimental distributed systems. Parallel Processing Letters **19**(03) (2009)
17. Li, Z., et al.: Cost and energy aware scheduling algorithm for scientific workflows with deadline constraint in clouds. IEEE Transactions on Services Computing **11**(4), 713–726 (2018)
18. Liu, H., et al.: Performance and energy modeling for live migration of virtual machines. Cluster computing **16**(2) (2013)
19. Orgerie, A.C., et al.: A survey on techniques for improving the energy efficiency of large-scale distributed systems. ACM Computing Surveys (CSUR) **46**(4) (2014)
20. Pietri, I., Sakellariou, R.: Energy-aware workflow scheduling using frequency scaling. In: International Conference on Parallel Processing Workshops (ICCPW) (2014)
21. Romanus, M., et al.: The anatomy of successful ECSS projects: Lessons of supporting high-throughput high-performance ensembles on XSEDE. In: XSEDE. pp. 1–9 (2012)
22. Samak, T., et al.: Energy consumption models and predictions for large-scale systems. In: International Parallel and Distributed Processing Symposium Workshops (IPDPSW). pp. 899–906 (2013)
23. Shepherd, D., et al.: Workflow scheduling on power constrained vms. In: IEEE/ACM 8th International Conference on Utility and Cloud Computing (2015)
24. Ferreira da Silva, R., et al.: Community resources for enabling and evaluating research on scientific workflows. In: IEEE International Conference on e-Science. eScience'14 (2014)
25. Ferreira da Silva, R., et al.: Online task resource consumption prediction for scientific workflows. Parallel Processing Letters **25**(3) (2015)
26. Taylor, I., et al.: Workflows for e-Science. Springer (2007)
27. Wang, L., et al.: Energy-aware parallel task scheduling in a cluster. Future Generation Computer Systems **29**(7) (2013)
28. The WRENCH Project. `http://wrench-project.org/` (2019)
29. WRENCH Pegasus Simulator. `https://github.com/wrench-project/pegasus` (2019)
30. Wu, T., et al.: Soft error-aware energy-efficient task scheduling for workflow applications in DVFS-enabled cloud. Journal of Systems Architecture **84**, 12–27 (2018)