

# DCWoRMS - a tool for simulation of energy efficiency in Grids and Clouds

Krzysztof Kurowski, Ariel Oleksiak, Wojciech Piatek, Tomasz Piontek,  
Andrzej Przybyszewski

---

## Abstract

In the recent years, the issue of computing infrastructures energy-efficiency has gained great attention. In this paper we present a Data Center Workload and Resource Management Simulator (DCWoRMS) which enables modeling and simulations of computing infrastructures to estimate their performance, energy consumption, and energy-efficiency metrics for diverse workloads and management policies. We discuss methods of power usage modeling available in the simulator. To this end, we compare the results of simulations to measurements from the real servers. To demonstrate DCWoRMS capabilities we evaluate the impact of several resource management policies on overall energy-efficiency of specific workloads on heterogeneous resources.

*Keywords:*

---

## 1. Introduction

TODO - update

Data centers are responsible for around 2% of the global energy consumption making it equal to the demand of aviation industry [13]. In many current data centers the actual IT equipment uses only half of the total energy (e.g. 45-62% in [12]) while most of the remaining part is required for cooling and air movement resulting in poor Power Usage Effectiveness (PUE) [22] values. Large energy needs and significant  $CO_2$  emissions caused that issues related to cooling, heat transfer, and IT infrastructure location are more and more carefully studied during planning and operation of data centers. Even if we take ecological and footprint issues aside, the amount of consumed energy can impose strict limits on data centers. First of all, energy bills may reach millions euros making computations expensive. Furthermore,

available power supply is usually limited so it also may reduce data center development capabilities, especially looking at challenges related to exascale computing breakthrough foreseen within this decade.

For these reasons many efforts were undertaken to measure and study energy efficiency of data centers. Some of projects focused on data center monitoring and management [7][1] whereas others on prototypes of low power computing infrastructures [21]. Studies included aspects such as energy efficiency of networks [5] and service level agreements related to energy consumption [9]. Additionally, vendors offer a wide spectrum of energy efficient solutions for computing and cooling [23][18][20]. However, a variety of solutions and configuration options can be applied planning new or upgrading existing data centers. In order to optimize a design or configuration of data center we need a thorough study using appropriate metrics and tools evaluating how much computation or data processing can be done within given power and energy budget and how it affects temperatures, heat transfers, and airflows within data center. Therefore, there is a need for simulation tools and models that approach the problem from a perspective of end users and take into account all the factors that are critical to understanding and improving the energy efficiency of data centers, in particular, hardware characteristics, applications, management policies, and cooling. These tools should support data center designers and operators by answering questions how specific application types, levels of load, hardware specifications, physical arrangements, cooling technology, etc. impact overall data center energy efficiency.

In this paper we present a Data Center Workload and Resource Management Simulator (DCWoRMS) which enables modeling and simulations of computing infrastructures to estimate their performance, energy consumption, and energy-efficiency metrics for diverse workloads and management policies. We discuss methods of power usage modeling available in the simulator. To this end, we compare results of simulations to measurements from the real servers. To demonstrate DCWoRMS capabilities we evaluate impact of several resource management policies on overall energy-efficiency of specific workloads on heterogeneous resources.

The remaining part of this paper is organized as follows. In Section 2 we give a brief overview of the current state of the art concerning modeling and simulation of distributed systems, like Grids and Clouds, in terms of energy efficiency. Section 3 discusses the main features of DCWoRMS. In particular, it introduces our approach to workload and resource management, presents

the concept of energy efficiency modeling and explains how to incorporate a specific application performance model into simulations. Section 4 discusses energy models adopted within the DCWoRMS. In Section 5 we present some experiments that were performed using DCWoRMS utilizing real testbed nodes models to show various types of popular resource and scheduling techniques allowing to decrease the total power consumption of the execution of a set of tasks. Section 6 focuses on the role of DCWoRMS within the CoolE-mAll project. Final conclusions and directions for future work are given in Section 7.

## 2. Related Work

The growing importance of energy-efficiency in information technologies led to significant interest in energy saving methods for computing systems. Nevertheless, studies of impact of resource management policies on energy-efficiency of IT infrastructures require a large effort and are difficult to perform in real distributed environments. To overcome these issues, extensive research has been conducted in the area of modeling and simulation and variety of tools that address the green computing have emerged. The most popular ones are: GreenCloud [8], CloudSim [2] and DCSG Simulator [3].

GreenCloud is a C++ based simulation environment for studying the energy-efficiency of cloud computing data centers. CloudSim is a simulation tool that allows modeling of cloud computing environments and evaluation of resource provisioning algorithms. Finally, the DCSG Simulator is a data center cost and energy simulator calculating the power and cooling schema of the data center equipment.

The scope of the aforementioned toolkits concerns the data center environments. However, all of them, except DCWoRMS presented in this paper, imposes and restricts user in terms of modeled resources. GreenCloud defines switches, links and servers that are responsible for task execution and may contain different scheduling strategies. Contrary to what the GreenCloud name may suggest, it does not allow testing the impact of a virtualization-based approaches. CloudSim allows creating a simple resources hierarchy consisting of machines and processors. To simulate a real cloud computing data center, it provides an extra virtualization layer responsible for the VM provisioning process and managing the VM life cycle. In DCSG Simulator user is able to take into account a variety of mechanical and electrical devices

as well as the IT equipment and define for each of them numerous factors, including device capacity and efficiency as well as the data center conditions.

The general idea behind all of the analyzed tools is to enable studies concerning energy efficiency in distributed infrastructures. GreenCloud approach enables simulation of energy usage associated with computing servers and network components. For example, the server power consumption model implemented in GreenCloud depends on the server state as well as its utilization. The CloudSim framework provides basic models to evaluate energy-conscious provisioning policies. Each computing node can be extended with a power model that estimates the current the power consumption. Within the DCSG Simulator, performance of each data center equipment (facility and IT) is determined by a combination of factors, including workload, local conditions, the manufacturer’s specifications and the way in which it is utilized. In DCWoRMS, the plugin idea has been introduced that offers emulating the behavior of computing resources in terms of power consumption. Additionally, it delivers detailed information concerning resource and application characteristics needed to define more sophisticated power draw models.

In order to emulate the behavior of real computing systems, green computing simulator should address also the energy-aware resource management. In this term, GreenCloud offers capturing the effects of both of the Dynamic Voltage and Frequency Scaling (DVFS) and Dynamic Power Management schemes. At the links and switches level, it supports downgrading the transmission rate and putting network equipment into a sleep mode. CloudSim comes with a set of predefined and extensible policies that manage the process of VM migrations in order to optimize the power consumption. However, the proposed approach is not sufficient for modeling more sophisticated policies like frequency scaling techniques and managing resource power states. Romonets tool is told to implement a set of basic energy-efficient rules that have been developed on the basis of detailed understanding of the data center as a system. The output of this simulation is a set of energy, like PUE, and cost data representing the IT devices. DCWoRMS introduces a dedicated interface that provides methods to obtain the detailed information about each resource and its components energy consumption and allows changing its current energy state. Availability of these interfaces in scheduling plugin supports implementation of various strategies such as centralized energy management, self-management of computing resources and mixed models.

In terms of application modeling, all tools, except DCSG Simulator, describe the application with a number of computational and communicational

requirements. In addition, GreenCloud and DCWoRMS allow introducing the QoS requirements (typical for cloud computing applications) by taking into account the time constraints during the simulation. DCSG Simulator instead of modeling of the single application, enables the definition of workload that leads to a given utilization level. However, only DCWoRMS supports application performance modeling by not only incorporating simple requirements that are taken into account during scheduling, but also by allowing specification of task execution time.

GreenCloud, CloudSim and DCWoRMS are released as Open Source under the GPL. Romonets tool is available under an OSL V3.0 open-source license, however, it can be only accessed by the DCSG members.

Summarizing, DCWoRMS stands out from other tools due to the flexibility in terms of data center equipment and structure definition. Moreover, it allows to associate the energy consumption not only with the current power state and resource utilization but also with the particular set of applications running on it. Moreover, it does not limit the user in defining various types of resource management policies. The main strength of CloudSim lies in implementation of the complex scheduling and task execution schemes involving resource virtualization techniques. However, the energy efficiency aspect is limited only to the VM management, The GreenCloud focuses on data center resources with particular attention to the network infrastructure and the most popular energy management approaches. DCSG simulator allows to take into account also non-computing devices, nevertheless it seems to be hardly customizable tool.

### 3. DCWoRMS

The following picture (Figure 1) presents the overall architecture of the simulation tool.

Data Center workload and resource management simulator (DCWoRMS) is a simulation tool based on the GSSIM framework [10] developed by Poznan Supercomputing and Networking Center (PSNC). GSSIM has been proposed to provide an automated tool for experimental studies of various resource management and scheduling strategies in distributed computing systems. DCWoRMS extends its basic functionality and adds some additional features related to the energy efficiency issues in data centers. In this section we will introduce the functionality of the simulator, in terms of modeling and simulation of large scale distributed systems like Grids and Clouds.

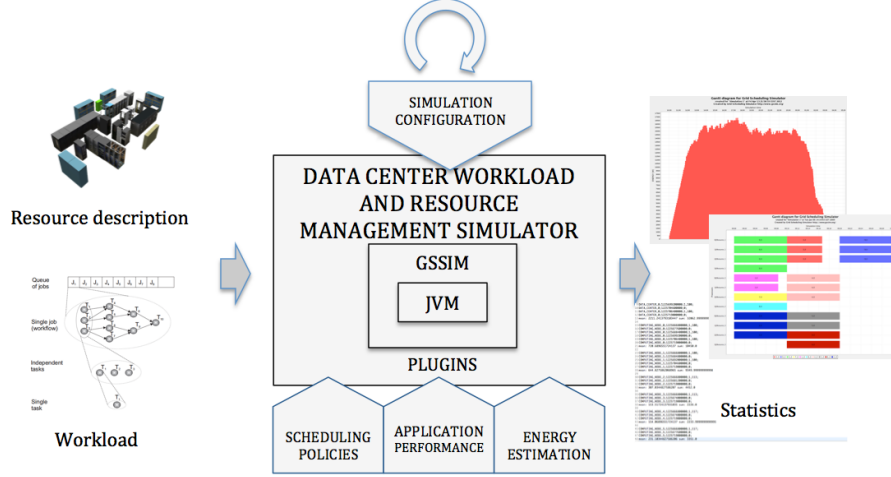


Figure 1: DCWoRMS architecture

### 3.1. Architecture

DCWoRMS is an event-driven simulation tool written in Java. In general, input data for the DCWoRMS consist of workload and resources descriptions. They can be provided by the user, read from real traces or generated using the generator module. However, the key elements of the presented architecture are plugins. They allow the researchers to configure and adapt the simulation environment to the peculiarities of their studies, starting from modeling job performance, through energy estimations up to implementation of resource management and scheduling policies. Each plugin can be implemented independently and plugged into a specific experiment. Results of experiments are collected, aggregated, and visualized using the statistics module. Due to a modular and plug-able architecture DCWoRMS can be applied to specific resource management problems and address different users requirements.

### 3.2. Workload modeling

As it was said, experiments performed in DCWoRMS require a description of applications that will be scheduled during the simulation. As a primary definition, DCWoRMS uses files in the Standard Workload Format (SWF) or its extension the Grid Workload Format (GWF) [14]. In addition to the SWF file, some more detailed specification of a job and tasks can be included in an auxiliary XML file. This form of description provides the scheduler

with more detailed information about application profile, task requirements, user preferences and execution time constraints, which are unavailable in SWF/GWF files. To facilitate the process of adapting the traces from real resource management systems, DCWoRMS supports reading those delivered from the most common ones like SLURM [15] and Torque [17]. Since the applications may vary depending on their nature in terms of their requirements and structure, DCWoRMS provides user flexibility in defining the application model. Thus, considered workloads may have various shapes and levels of complexity that range from multiple independent jobs, through large-scale parallel applications, up to whole workflows containing time dependencies and preceding constraints between jobs and tasks. Each job may consist of one or more tasks and these can be seen as groups of processes. Moreover, DCWoRMS is able to handle rigid and moldable jobs, as well as pre-emptive ones. To model the application profile in more detail, DCWoRMS follows the DNA approach proposed in [6]. Accordingly, each task can be presented as a sequence of phases, which shows the impact of this task on the resources that run it. Phases are then periods of time where the system is stable (load, network, memory) given a certain threshold. Each phase is linked to values of the system that represent a resource consumption profile. Such a stage could be for example described as follows: 60% CPU, 30% net, 10% mem.

Levels of information about incoming jobs are presented in Figure 2.

This form of representation allows users to define a wide range of workloads: HPC (long jobs, computational-intensive, hard to migrate) or virtualization (short requests) typical for cloud computing environments. Further, the DCWoRMS benefits from the GSSIM workload generator tool and extends it with that allows creating synthetic workloads.

### 3.3. Resource modeling

The main goal of DCWoRMS is to enable researchers evaluation of various resource management policies in diverse computing environments. To this end, it supports flexible definition of simulated resources both on physical (computing resources) as well as on logical (scheduling entities) level. This flexible approach allows modeling of various computing entities consisting of compute nodes, processors and cores. In addition, detailed location of the given resources can be provided in order to group them and arrange into physical structures such as racks and containers. Each of the components may be described by different parameters specifying available memory,

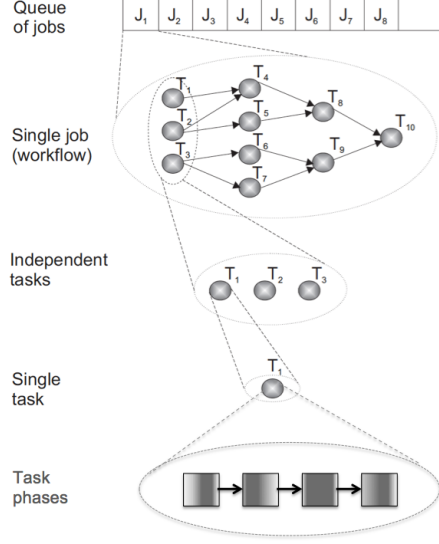


Figure 2: Levels of information about jobs

storage capabilities, processor speed etc. In this way, it is possible to describe power distribution system and cooling devices. Due to an extensible description, users are able to define a number of experiment-specific and visionary characteristics. Moreover, with every component, dedicated profiles can be associated that determines, among others, power, thermal and air throughput properties. The energy estimation plugin can be bundled with each resource. This allows defining various power models that can be then followed by different computing system components. Details concerning the approach to energy-efficiency modeling in DCWoRMS can be found in the next sections.

Scheduling entities allow providing data related to the brokering or queuing system characteristics. Thus, information about available queues, resources associated with them and their parameters like priority, availability of AR mechanism etc. can be defined. Moreover, allocation policy and task scheduling strategy for each scheduling entity can be introduced in form of the reference to an appropriate plugin. DCWoRMS allows building a hierarchy of schedulers corresponding to the hierarchy of resource components over which the task may be distributed.

In this way, the DCWoRMS supports simulation of a wide scope of physi-

cal and logical architectural patterns that may span from a single computing resource up to whole data centers or geographically distributed grids and clouds. In particular, it supports simulating complex distributed architectures containing models of the whole data centers, containers, racks, nodes, etc. In addition, new resources and distributed computing entities can easily be added to the DCWoRMS environment in order to enhance the functionality of the tool and address more sophisticated requirements. Granularity of such topologies may also differ from coarse-grained to very fine-grained modeling single cores, memory hierarchies and other hardware details.

### *3.4. Energy management concept in DCWoRMS*

The DCWoRMS allows researchers to take into account energy efficiency and thermal issues in distributed computing experiments. That can be achieved by the means of appropriate models and profiles. In general, the main goal of the models is to emulate the behavior of the real computing resources, while profiles support models by providing data essential for the power consumption calculations. Introducing particular models into the simulation environment is possible through choosing or implementation of dedicated energy plugins that contain methods to calculate power usage of resources, their temperature and system air throughput values. Presence of detailed resource usage information, current resource energy and thermal state description and a functional energy management interface enables an implementation of energy-aware scheduling algorithms. Resource energy consumption and thermal metrics become in this context an additional criterion in the resource management process. Scheduling plugins are provided with dedicated interfaces, which allow them to collect detailed information about computing resource components and to affect their behavior. The following subsections present the general idea behind the energy-efficiency simulations.

#### *3.4.1. Power management*

The motivation behind introducing a power management concept in DCWoRMS is providing researchers with the means to define the energy efficiency of resources, dependency of energy consumption on resource load and specific applications, and to manage power modes of resources. Proposed solution extends the power management concept presented in GSSIM [11] by offering a much more granular approach with the possibility of plugging energy consumption models and power profiles into each resource level.

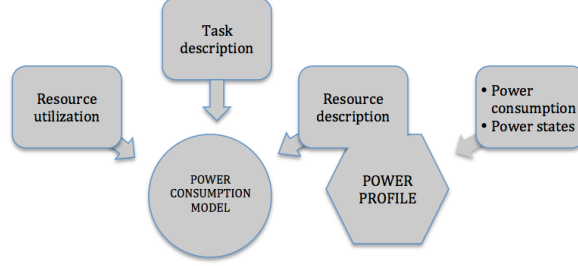


Figure 3: Power consumption modeling

**Power profile.** In general, power profiles allow specifying the power usage of resources. Depending on the accuracy of the model, users may provide additional information about power states which are supported by the resources, amounts of energy consumed in these states, and other information essential to calculate the total energy consumed by the resource during runtime. In such a way each component of IT infrastructure may be described, including computing resources, system components and data center facilities. Moreover, it is possible to define any number of new, resource specific, states, for example so called P-states, in which processor can operate.

**Power consumption model.** The main aim of these models is to emulate the behavior of the real computing resource and the way it consumes energy. Due to a rich functionality and flexible environment description, DCWoRMS can be used to verify a number of theoretical assumptions and to develop new energy consumption models. Modeling of energy consumption is realized by the energy estimation plugin that calculates energy usage based on information about the resource power profile, resource utilization, and the application profile including energy consumption and heat production metrics. Relation between model and power profile is illustrated in Figure 3.

**Power management interface.** DCWoRMS is complemented with an interface that allows scheduling plugins to collect detailed power information about computing resource components and to change their power states. It enables performing various operations on the given resources, including dynamically changing the frequency level of a single processor, turning off/on computing resources etc. The activities performed with this interface find a reflection in total amount of energy consumed by the resource during simulation.

Presence of detailed resource usage information, current resource energy state description and functional energy management interface enables an implementation of energy-aware scheduling algorithms. Resource energy consumption becomes in this context an additional criterion in the scheduling process, which use various techniques to decrease energy consumption, e.g. workload consolidation, moving tasks between resources to reduce a number of running resources, dynamic power management, cutting down CPU frequency, and others.

#### *3.4.2. Air throughput management concept*

The presence of an air throughput concept addresses the issue of resource air-cooling facilities provisioning. Using the air throughput profiles and models allows anticipating the air flow level on output of the computing system component, resulting from air-cooling equipment management.

***Air throughput profile.*** The air throughput profile, analogously to the power profile, allows specifying supported air flow states. Each air throughput state definition consists of an air flow value and a corresponding power draw. It can represent, for instance, a fan working state. In this way, associating the air throughput profile with the given computing resource, it is possible to describe mounted air-cooling devices. Possibility of introducing additional parameters makes the air throughput description extensible for new specific characteristics.

***Air throughput model.*** Similar to energy consumption models, the user is provided with a dedicated interface that allows him to describe the resulting air throughput of the computing system components like cabinets or server fans. The general idea of the air throughput modeling is shown in Figure 4. Accordingly, air flow estimations are based on detailed information about the involved resources, including their air throughput states.

***Air throughput management interface.*** The DCWoRMS delivers interfaces that provide access to the air throughput profile data, allows acquiring detailed information concerning current air flow conditions and changes in air flow states. The availability of these interfaces support evaluation of different cooling strategies.

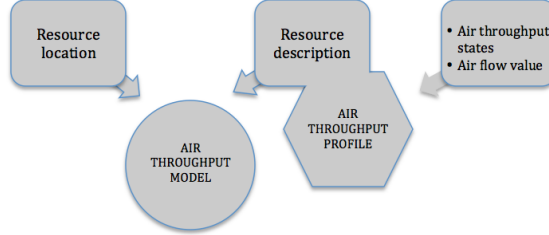


Figure 4: Air throughput modeling

### 3.4.3. Thermal management concept

The primary motivation behind the incorporation of thermal aspects in the DCWoRMS is to exceed the commonly adopted energy use-cases and apply more sophisticated scenarios. By the means of dedicated profiles and interfaces, it is possible to perform experimental studies involving temperature-aware workload placement.

**Thermal profile.** Thermal profile expresses the thermal specification of resources. It consists of the definition of the thermal design power (TDP), thermal resistance and thermal states that describe how the temperature depends on dissipated heat. For the purposes of more complex experiments, introducing of new, user-defined characteristics is supported. The aforementioned values may be provided for all computing system components distinguishing them, for instance, according to their material parameters and/or models.

**Temperature estimation model.** Thermal profile, complemented with the temperature measurement model implementation may introduce temperature sensors simulation. In this way, users have means to approximately predict the temperature of the simulated objects by taking into account basic thermal characteristics as well as the estimated impact of cooling devices. However, the proposed approach assumes some simplifications that ignore heating and cooling dynamics understood as a heat flow process.

Figure 8 summarizes relation between model and profile and input data.

**Thermal resource management interface.** As the temperature is highly dependent on the dissipated heat and cooling capacity, thermal resource management is performed via a power and air throughput interface. Nevertheless,

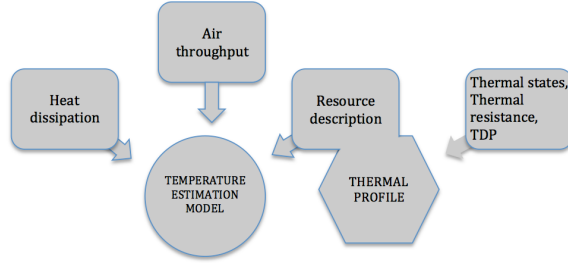


Figure 5: Temperature estimation modeling

the interface provides access to the thermal resource characteristics and the current temperature values

### 3.5. Application performance modeling

In general, DCWoRMS implements user application models as objects describing computational, communicational as well as energy requirements and profiles of the task to be scheduled. Additionally, simulator provides means to include complex and specific application performance models during simulations. They allow researchers to introduce specific ways of calculating task execution time. These models can be plugged into the simulation environment through a dedicated API and implementation of an appropriate plugin. To specify the execution time of a task user can apply a number of parameters, including:

- task length (number of CPU instructions)
- task requirements
- detailed description of allocated resources (processor type and parameters, available memory)
- input data size
- network parameters

Using these parameters developers can for instance take into account the architectures of the underlying systems, such as multi-core processors, or virtualization overheads, and their impact on the final performance of applications.

#### 4. Modeling of energy efficiency in DCWoRMS

DCWoRMS is an open framework in which various models and algorithms can be investigated as presented in Section 3.5. We discuss possible approaches to modeling that can be applied to simulation of energy-efficiency of distributed computing systems in this section. Additionally, to facilitate the simulation process, DCWoRMS provides some basic implementation of power consumption, air throughput and thermal models. We described them as examples and validate part of them by experiments in real computing system (in Section 5).

The most common questions explored by researchers who study energy-efficiency of distributed computing systems is how much energy  $E$  do these systems require to execute workloads. In order to obtain this value the simulator must calculate values of power  $P_i(t)$  and load  $L_i(t)$  in time for all  $m$  computing nodes,  $i = 1..m$ . Load function may depend on specific load models applied. In more complex cases it can even be defined as vectors of different resource usage in time. In a simple case load can be either idle or busy but even in this case estimation of job processing times  $p_j$  is needed to calculate total energy consumption. The total energy consumption of computing nodes is given by (1):

$$E = \sum_i^m \int_t P_i(t) \quad (1)$$

Power function may depend on load and states of resources or even specific applications as explained in 4.1. Total energy can be also completed by adding constant power usage of components that does not depend on load or state of resources.

In large computing systems which are often characterized by high computational density, total energy consumption of computing nodes is not the only result interesting for researchers. Temperature distribution is getting more and more important as it affects energy consumption of cooling devices, which can reach even half of a total data center energy use. In order to obtain accurate values of temperatures heat transfer simulations based on the Computational Fluid Dynamics (CFD) methods have to be performed. These methods require as an input (i.e. boundary conditions) a heat dissipated by IT hardware and air throughput generated by fans at servers' outlets. Another approach is based on simplified thermal models that without costly CFD calculations provide rough estimations of temperatures. DCWoRMS

enables the use of either approaches. In the the former, the output of simulations including power usage of computing nodes in time and air throughput at node outlets can be passed to CFD solver. This option is further elaborated in Section 6. Simplified thermal models required by the latter approach are proposed in 4.3.

#### 4.1. Power consumption models

As stated above power usage of computing nodes depend on a number of factors.

Generally, the power consumption of a modern CPU is given by the Ohm's law:

$$P = C \cdot V_{core}^2 \cdot f \quad (2)$$

with  $C$  being the processor switching capacitance,  $V_{core}$  the current P-State's core voltage and  $f$  the frequency. Based on the above equation it is suggested that although the reduction of frequency causes an increase in the time of execution, the reduction of frequency also leads to the reduction of  $V_{core}$  and thus the power savings from the  $P \sim V_{core}^2$  relation outweigh the increased computation time. However, experiments performed on several HPC servers shown that this dependency does not reflect theoretical shape and is often close to linear as presented in Figure 4.1. This phenomenon can be explained by impact of other component than CPU and narrow range of available voltages. A good example of impact by other components is power usage of servers with visible influence of fans as illustrated in Figure 7.

For these reasons, DCWoRMS allows users to define dependencies between power usage and resource states (such as CPU frequency) in the form of tables or arbitrary functions using energy estimation plugins.

The energy consumption models provided by default can be classified into the following groups, starting from the simplest model up to the more complex ones. Users can easily switch between the given models and incorporate new, visionary scenarios.

**Static approach** is based on a static definition of resource power usage. This model calculates the total amount of energy consumed by the computing resource system as a sum of energy, consumed by all its components (processors, disks, power adapters, etc.). More advanced versions of this approach assume definition of resource states along with corresponding power usage. This model follows changes of resource power states and sums

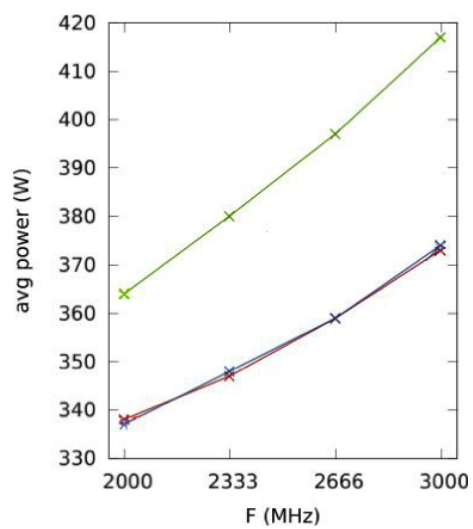


Figure 6: Average power usage with regard to CPU frequency  
**Tests:** Linpack (*green*), Abinit (*purple*), Namd (*blue*) and Cpuburn (*red*).

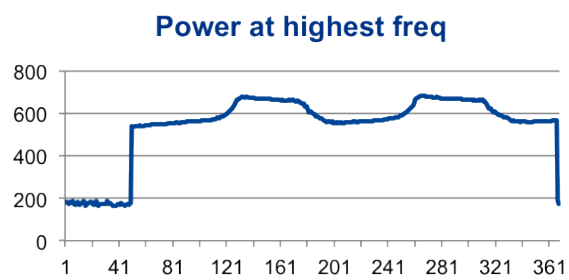


Figure 7: Power in time for highest frequency

up the amounts of energy defined for each state. In this case, specific values of power usage are defined for all discrete  $n$  states as shown in (3):

$$S_1 \rightarrow P_1, S_2 \rightarrow P_2, \dots, S_n \rightarrow P_n \quad (3)$$

**Resource load** model extends the static power state description and enhances it with real-time resource usage, most often simply the processor load. In this way it enables a dynamic estimation of power usage based on resource basic power usage and state (defined by the static resource description) as well as resource load. For instance, it allows distinguishing between the amount of energy used by idle processors and processors at full load. In this manner, energy consumption is directly connected with power state and describes average power usage by the resource working in a current state. In this case, specific values of power usage are defined for all pairs state and load values (discretized to  $l$  values) as shown in (4):

$$(S_1, L_1) \rightarrow P_{11}, (S_1, L_2) \rightarrow P_{12}, \dots, (S_2, L_1) \rightarrow P_{21}, \dots, (S_n, L_l) \rightarrow P_{nl}, \quad (4)$$

**Application specific** model allows expressing differences in the amount of energy required for executing various types of applications at diverse computing resources. It considers all defined system elements (processors, memory, disk, etc.), which are significant in total energy consumption. Moreover, it also assumes that each of these components can be utilized in a different way during the experiment and thus have different impact on total energy consumption. To this end, specific characteristics of resources and applications are taken into consideration. Various approaches are possible including making the estimated power usage dependent on defined classes of applications, ratio between CPU-bound and IO-bound operations, etc. In this case, power usage is an arbitrary function of state, load, and application characteristics as shown in (5):

$$f(S, L, A) \rightarrow P \quad (5)$$

#### 4.2. Air throughput models

The DCWoRMS comes with the following air throughput models. By default, air throughput estimations are performed according to the first one.

**Static** model refers to a static definition of air throughput states. According to this approach, output air flow depends only on the present air cooling

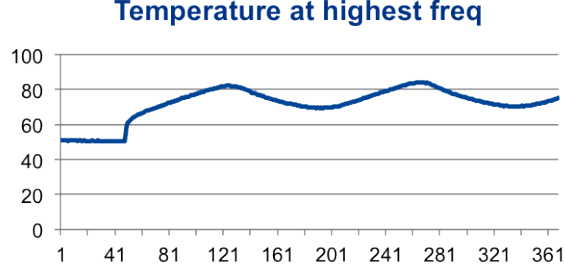


Figure 8: Temperature in time for highest frequency

working state and the corresponding air throughput value. Each state change triggers the calculations and updates the current air throughput value. This strategy requires only a basic air throughput profile definition.

**Space** model allows taking into account a duct associated with the investigated air flow. On the basis of the given fan rotation speed and the obstacles before/behind the fans, the output air throughput can be roughly estimated. To this end, additional manufacturer’s specifications will be required, including resulting air velocity values and fan duct dimensions. Thus, it is possible to estimate the air flow level not only referring to the current fan operating state but also with respect to the resource and its subcomponent placement. More advanced scenario may consider mutual impact of several air flows.

#### 4.3. Thermal models

The following models are supported natively. By default, the static strategy is applied.

**Static** approach follows the changes in heat, generated by the computing system components and matches the corresponding temperature according to the specified profile. Since it tracks the power consumption variations, corresponding values must be delivered, either from power consumption model or on the basis of user data. Replacing the appropriate temperature values with function based on the defined material properties and/o experimentally measured values can easily extend this model.

**Ambient** model allows taking into account the surrounding cooling infrastructure. It calculates the device temperature as a function adopted from the static approach and extends it with the influence of cooling method. The

efficiency of cooling system may be derived from the current air throughput value.

## 5. Experiments and evaluation

TODO - correct, improve, refactor...

In this section, we present computational analysis that were conducted to emphasize the role of modelling and simulation in studying computing systems performance. To this end we evaluate the impact of energy-aware resource management policies on overall energy-efficiency of specific workloads on heterogeneous resources. The following sections contain description of the used system, tested application and the results of simulation experiments conducted for the evaluated strategies.

### 5.1. Testbed description

To obtain values of power consumption that could be later used in DC-WoRMS environment to build the model and to evaluate resource management policies we ran a set of applications / benchmarks on the physical testbed. For experimental purposes we choose the high-density Resource Efficient Cluster Server (RECS) system. The single RECS unit consists of 18 single CPU modules, each of them can be treated as an individual node of PC class. Configuration of our RECS unit is presented in Table 1.

Nodes		
Type	Memory (RAM)	Count
Intel i7	16 GB	8
AMD Fusion T40N 64 Bit	4 GB	6
Atom D510 64 Bit	2 GB	4

Table 1: RECS system configuration

The RECS system was chosen due to its heterogeneous platform with very high density and energy efficiency that has a monitoring and controlling mechanism integrated. The built-in and additional sensors allow monitoring the complete testbed at a very fine granularity level without the negative impact of the computing- and network-resources.

### 5.2. Evaluated applications

As mentioned, first we carried out a set of tests on the real hardware used as a CoolEmAll testbed to build the performance and energy profile of applications. Then we applied this data into the simulation environment and used to investigate different approaches to energy-aware resource management. The following applications were evaluated:

**Abinit** is a widely-used application for computational physics simulating systems made of electrons and nuclei to be calculated within density functional theory.

**C-Ray** is a ray-tracing benchmark that stresses floating point performance of a CPU. Our test is configured with the 'scene' file at 16000x9000 resolution.

**Linpack** benchmark is used to evaluate system floating point performance. It is based on the Gaussian elimination methods that solve a dense  $N$  by  $N$  system of linear equations.

**Tar** it is a widely used data archiving software]. In our tests the task was to create one compressed file of Linux kernel (version 3.4), which is about 2,3 GB size, using bzip2.

**FFTE** benchmark measures the floating-point arithmetic rate of double precision complex one-dimensional Discrete Fourier Transforms of 1-, 2-, and 3-dimensional sequences of length  $2^p * 3^q * 5^r$ . In our tests only one core was used to run the application.

### 5.3. Methodology

Every chosen application / benchmark was executed on each type of node, for all frequencies supported by the CPU and for different levels of parallelization (number of cores). To eliminate the problem with assessing which part of the power consumption comes from which application, in case when more then one application is ran on the node, the queuing system (SLURM) was configured to run jobs in exclusive mode (one job per node). Such configuration is often used for at least dedicated part of HPC resources. The advantage of the exclusive mode scheduling policy consist in that the job gets all the resources of the assigned nodes for optimal parallel performance and applications running on the same node do not influence each other. For every configuration of application, type of node and CPU frequency we measure the average power consumption of the node and the execution time. The aforementioned values were used to configure the DCWoRMS environment providing energy and time execution models. Based on the models obtained

Characteristic	Load intensity				Distribution
	10	30	50	70	
Task Count	1000				constant
Task Interval [s]	3000	1200	720	520	poisson
Number of cores to run		1			uniform - 30%
		2			uniform - 30%
		3			uniform - 10%
		4			uniform - 10%
		5			uniform - 5%
		6			uniform - 5%
		7			uniform - 5%
		8			uniform - 5%
Application type		Abinit			uniform - 20%
		C-Ray			uniform - 20%
		Tar			uniform - 20%
		Linpack - 3Gb			uniform - 10%
		Linpack - tiny			uniform - 10%
		FFT			uniform - 20%

Table 2: Workload characteristics

for the considered set of resources and applications we evaluated a set of resource management strategies in terms of energy consumption needed to execute three workloads varying in load intensity (10%, 30%, 50%, 70%). To generate a workload we used the DCWoRMS workload generator tool using the following characteristics.

In all cases we assumed that tasks are scheduled and served in order of their arrival (FIFO strategy with easy backfilling strategy).

#### 5.4. Computational analysis

In the following section we present the results obtained for the workload with load density equal to 70% in the light of five resource management and scheduling strategies. The scheduling strategies were evaluated according to two criteria: total energy consumption and maximum completion time of all tasks (makespan). In the second part of this section we discuss the corresponding results received for workloads with other density level.

#### 5.4.1. Random approach

The first considered by us policy was the Random strategy in which tasks were assigned to nodes in random manner with the reservation that they can be assigned only to nodes of the type which the application was possible to execute on and we have the corresponding value of power consumption and execution time. The Random strategy is only the reference one and will be later used to compare benefits in terms of energy efficiency resulting from more sophisticated algorithms. Criteria values are as follows: **total energy usage**: 46,883 kWh, **workload completion time**: 533 820 s. Figure 9 presents the energy consumption, load of the system and obtained schedule, respectively.

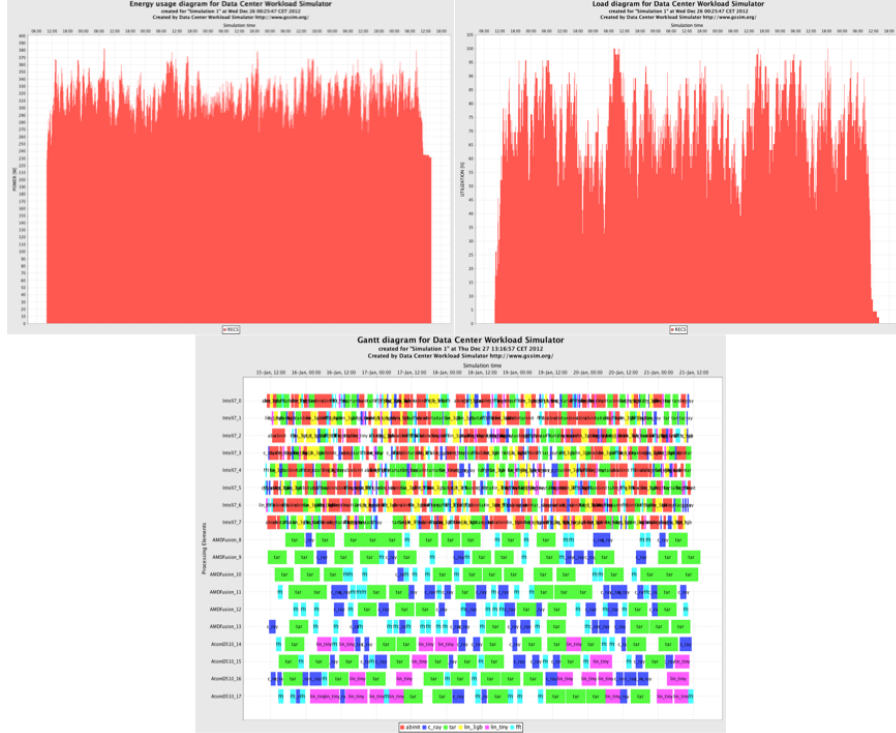


Figure 9: Random strategy

In the second version of this strategy, which is getting more popular due to energy costs, we switched off unused nodes to reduce the total energy consumption. In the previous one, unused nodes are not switched off, which case is still the primary one in many HPC centers.

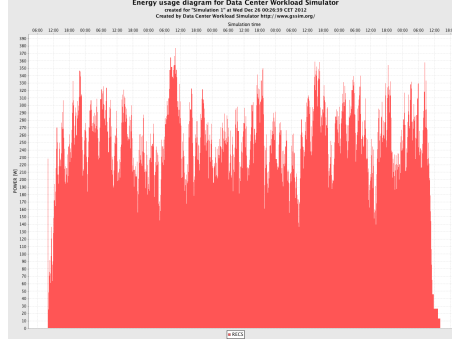


Figure 10: Random + switching off unused nodes strategy

In this version of experiment we neglected additional cost and time necessary to change the power state of resources. As can be observed in the power consumption chart in the Figure 10, switching of unused nodes led to decrease of the total energy consumption. As expected, with respect to the makespan criterion, both approaches perform equally reaching **workload completion time**: 533 820 s. However, the pure random strategy was significantly outperformed in terms of energy usage, by the policy with additional node power management with its **total energy usage**: 36,705 kWh. The overall energy savings reached 22%.

#### 5.4.2. Energy optimization

The next two evaluate resource management strategies try to decrease the total energy consumption needed to execute the whole workload taking into account differences in applications and hardware profiles. We tried to match both profiles to find the more energy efficient assignment. In the first case we assumed that there is again no possibility to switch off unused nodes, thus for the whole time needed to execute workload nodes consume at least power for idle state. To obtain the minimal energy consumption, tasks has to be assigned to the nodes of type for which the difference between energy consumption for the node running the application and in the idle state is minimal. The power usage measured in idle state for three types of nodes is gathered in the Table 3.

As mentioned, we assign tasks to nodes minimizing the value of expression:  $(P - P_{idle}) * exec\_time$ , where  $P$  denotes observed power of the node running the particular application and  $exec\_time$  refers to the measured application running time. Based on the application and hardware profiles, we

Type	Power usage in idle state [W]
Intel i7	11,5
AMD Fusion	19
Atom D510	10

Table 3: Measured power of testbed nodes in idle state

expected that Atom D510 would be the preferred node. Obtained schedule, that is presented in the Gantt chart in Figure 11 along with the energy and system usage, confirmed our assumptions. Atom D510 nodes are nearly fully loaded, while the least energy-favourable AMD nodes are used only when other ones are busy.



Figure 11: Energy usage optimization strategy

This allocation strategy, leads to slight deterioration of makespan criterion, resulting in **workload completion time** equal to 534 400 s. Nevertheless, the **total energy usage** is reduced, achieving: 46,305 kWh.

The next strategy is similar to the previous one, so making the assignment of task to the node, we still take into consideration application and hardware profiles, but in that case we assume that the system supports possibility of switching off unused nodes. In this case the minimal energy consumption is achieved by assigning the task to the node for which the product of power consumption and time of execution is minimal. In other words we minimized the following expression:  $P * exec.time$ . Contrary to the previous strategy, we expected Intel I7 nodes to be allocated first. Generated Gantt chart is consistent with our expectations.

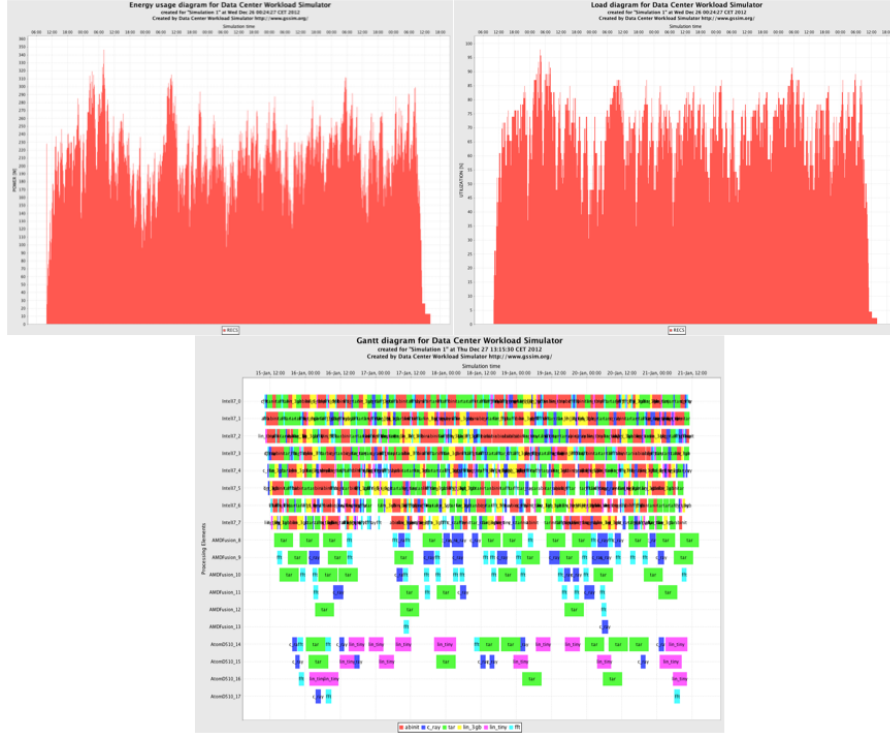


Figure 12: Energy usage optimization + switching off unused nodes strategy

Estimated **total energy usage** of the system is 30,568 kWh. As we can see, this approach significantly improved the value of this criterion, comparing to the previous policies. Moreover, the proposed allocation strategy does not worsen the **workload completion time** criterion, for which the resulting value is equal to 533 820 s.

### 5.4.3. Frequency scaling

The last considered by us case is modification of the random strategy. We assume that tasks do not have deadlines and the only criterion which is taken into consideration is the total energy consumption. In this experiment we configured the simulated infrastructure for the lowest possible frequencies of CPUs. The experiment was intended to check if the benefit of running the workload on less power-consuming frequency of CPU is not leveled by the prolonged time of execution of the workload. The values of the evaluated criteria are as follows: **workload completion time**: 1 065 356 s and **total energy usage**: 77,109 kWh. As we can see, for the given load of the system (70%), the cost of running the workload that requires almost twice more time, can not be compensate by the lower power draw. Moreover, as it can be observed on the charts in Figure 13 the execution times on the slowest nodes (Atom D510) visibly exceeds the corresponding values on other servers



Figure 13: Frequency downgrading strategy

As we were looking for the trade-off between total completion time and

energy usage, we were searching for the workload load level that can benefit from the lower system performance in terms of energy-efficiency. For the frequency downgrading policy, we noticed the improvement on the energy usage criterion only for the workload resulting in 10% system load. For this threshold we observed that slowdown in task execution does not affect the subsequent tasks in the system and thus total completion time of the whole workload.

Figure 14 shows schedules obtained for Random and DFS strategy. One should easily note that the

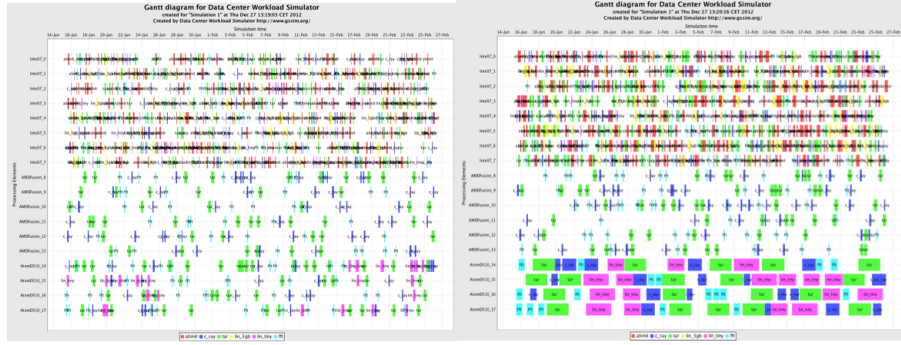


Figure 14: Schedules obtained for Random strategy (left) and DFS strategy (right) for 10% of system load

The following tables: Table 4 and Table 5 contain the values of evaluation criteria (total energy usage and makespan respectively) gathered for all investigated workloads.

Load	Strategy				
	R	R+NPM	EO	EO+NPM	DFS
10%	241,337	37,811	239,667	25,571	239,278
30%	89,853	38,059	88,823	25,595	90,545
50%	59,112	36,797	58,524	26,328	76,020
70%	46,883	36,705	46,305	30,568	77,109

Table 4: Energy usage [kWh] for different level of system load. R - Random, R+NPM - Random + node power management, EO - Energy optimization, EO+NPM - Energy optimization + node power management, DFS - Dynamic Frequency Scaling

Referring to the Table 4, one should easily note that gain from switching off unused nodes decreases with the increasing workload density. In general,

Load	Strategy				
	R	R+NPM	EO	EO+NPM	DFS
10%	3 605 428	3 605 428	3 605 428	3 605 428	3 622 968
30%	1 214 464	1 214 464	1 215 044	1 200 807	1 275 093
50%	729 066	729 066	731 126	721 617	1 049 485
70%	533 820	533 820	534 400	533 820	1 065 356

Table 5: Makespan [s] for different level of system load. R - Random, R+NPM - Random + node power management, EO - Energy optimization, EO+NPM - Energy optimization + node power management, DFS - Dynamic Frequency Scaling

for the highly loaded system such policy does not find an application due to the cost related to this process and relatively small benefits.

#### 5.4.4. Summary

### 6. DCWoRMS application/use cases

DCWoRMS in CoolEmAll, integration with CFD

...

Being based on the GSSIM framework, that has been successfully applied in a substantial number of research projects and academic studies, DCWoRMS with its sophisticated energy extension has become an essential tool for studies of energy efficiency in distributed environments. For this reason, it has been adopted within the CoolEmAll project as a component of Simulation, Visualisation and Decision Support (SVD) Toolkit. In general the main goal of CoolEmAll is to provide advanced simulation, visualisation and decision support tools along with blueprints of computing building blocks for modular data centre environments. Once developed, these tools and blueprints should help to minimise the energy consumption, and consequently the CO2 emissions of the whole IT infrastructure with related facilities. The SVD Toolkit is designed to support the analysis and optimization of IT modern infrastructures. For the recent years the special attention has been paid for energy utilized by the data centers which considerable contributes to the data center operational costs. Actual power usage and effectiveness of energy saving methods heavily depends on available resources, types of applications and workload properties. Therefore, intelligent resource management policies are gaining popularity when considering the energy efficiency of IT infrastructures. Hence, SVD Toolkit integrates also

workload management and scheduling policies to support complex modeling and optimization of modern data centres.

The main aim of DCWoRMS within CoolEmAll project is to enable studies of dynamic states of IT infrastructures, like power consumption and air throughput distribution, on the basis of changing workloads, resource model and energy-aware resource management policies. In this context, DCWoRMS takes into account the specific workload and application characteristics as well as detailed resource parameters. It will benefit from the CoolEmAll benchmarks and classification of applications and workloads. In particular various types of workload, including data centre workloads using virtualization and HPC applications, may be considered. The knowledge concerning their performance and properties as well as information about their energy consumption and heat production will be used in simulations to study their impact on thermal issues and energy efficiency. Detailed resource characteristics, will be also provided according to the CoolEmAll blueprints. Based on this data, workload simulation will support evaluation process of various resource management approaches. These policies may include a wide spectrum of energy-aware strategies such as workload consolidation/migration, dynamic switching off nodes, DVFS and thermal-aware methods. In addition to typical approaches minimizing energy consumption, policies that prevent too high temperatures in the presence of limited cooling (or no cooling) may also be analyzed. Moreover, apart from the set of predefined strategies, new approaches can easily be applied and examined. The outcome of the workload and resource management simulation phase is a distribution of power usage and air throughput for the computing models specified within the SVD Toolkit. These statistics may be analyzed directly by data centre designers and administrators and/or provided as an input to the CFD simulation phase. The former case allows studying how the above metrics change over time, while the latter harness CFD simulations to identify temperature differences between the computing modules, called hot spots. The goal of this scenario is to visualise the behavior of the temperature distribution within a server room with a number of racks for different types of executed workloads and for various policies used to manage these workloads.

## **7. Conclusions and future work**

TODO - Conclusions and future research

## References

- [1] A. Berl, E. Gelenbe, M. di Girolamo, G. Giuliani, H. de Meer, M.-Q. Dang, K. Pentikousis. Energy-Efficient Cloud Computing. *The Computer Journal*, 53(7), 2010.
- [2] Rodrigo N. Calheiros, Rajiv Ranjan, Anton Beloglazov, Cesar A. F. De Rose, and Rajkumar Buyya, CloudSim: A Toolkit for Modeling and Simulation of Cloud Computing Environments and Evaluation of Resource Provisioning Algorithms, *Software: Practice and Experience (SPE)*, Volume 41, Number 1, Pages: 23-50, ISSN: 0038-0644, Wiley Press, New York, USA, January, 2011.
- [3] <http://dcsg.bcs.org/welcome-dcsg-simulator>
- [4] <http://www.datacenterdynamics.com/blogs/ian-bitterlin/it-does-more-it-says-tin%E2%80%A6>
- [5] E. Gelenbe and C. Morfopoulou. Power savings in packet networks via optimised routing. *Mobile Networks and Applications*, 17(1):152159, February 2012.
- [6] Ghislain Landry Tsafack Chetsa, Laurent Lefvre, Jean-Marc Pierson, Patricia Stolf, Georges Da Costa. DNA-inspired Scheme for Building the Energy Profile of HPC Systems. In: *International Workshop on Energy-Efficient Data Centres*, Madrid, Springer, 2012
- [7] A. Kipp, L. Schubert, J. Liu, T. Jiang, W. Christmann, M. von dem Berge (2011). Energy Consumption Optimisation in HPC Service Centres, *Proceedings of the Second International Conference on Parallel, Distributed, Grid and Cloud Computing for Engineering*, B.H.V. Topping and P. Ivanyi, (Editors), Civil-Comp Press, Stirlingshire, Scotland
- [8] D. Kliazovich, P. Bouvry, and S. U. Khan, A Packet-level Simulator of Energy-aware Cloud Computing Data Centers, *Journal of Supercomputing*, vol. 62, no. 3, pp. 1263-1283, 2012
- [9] S. Klingert, T. Schulze, C. Bunse. GreenSLAs for the Energy-efficient Management of Data Centres. *2nd International Conference on Energy-Efficient Computing and Networking (e-Energy)*, 2011.

- [10] S. Bak, M. Krystek, K. Kurowski, A. Oleksiak, W. Piatek and J. Weglarz, GSSIM - a Tool for Distributed Computing Experiments, Scientific Programming Journal, vol. 19, no. 4, pp. 231-251, 2011.
- [11] M. Krystek, K. Kurowski, A. Oleksiak, W. Piatek, Energy-aware simulations with GSSIM. Proceedings of the COST Action IC0804 on Energy Efficiency in Large Scale Distributed Systems, 2010, pp. 55-58.
- [12] Hintemann, R., Fichter, K. (2010). Materialbestand der Rechenzentren in Deutschland, Eine Bestandsaufnahme zur Ermittlung von Ressourcen- und Energieeinsatz, UBA, Texte, 55/2010
- [13] Koomey, Jonathan. 2008. "Worldwide electricity used in data centers." Environmental Research Letters. vol. 3, no. 034008. September 23
- [14] <http://gwa.ewi.tudelft.nl/>
- [15] <https://computing.llnl.gov/linux/slurm/>
- [16] Parallel Workload Archive, <http://www.cs.huji.ac.il/labs/parallel/workload/>
- [17] <http://www.adaptivecomputing.com/products/open-source/torque/>
- [18] Colt Modular Data Centre, <http://www.colt.net/uk/en/products-services/data-centre-services/modular-data-centre-en.htm>
- [19] The CoolEmAll project website, <http://coolemall.eu>
- [20] EcoCooling, <http://www.ecocooling.org>
- [21] The MontBlanc project website, <http://www.montblanc-project.eu/>
- [22] The Green Grid Data Center Power Efficiency Metrics: PUE and DCiE, <http://www.thegreengrid.org/Global/Content/white-papers/The-Green-Grid-Data-Center-Power-Efficiency-Metrics-PUE-and-DCiE>
- [23] SGI ICE Cube Air, [http://www.sgi.com/products/data\\_center/ice\\_cube\\_air/](http://www.sgi.com/products/data_center/ice_cube_air/)