Security Supportive Energy Aware Scheduling and Scaling for Cloud Environments

Agnieszka Jakóbik, Daniel Grzonka
Institute of Computer Science
Cracow University of Technology
Warszawska st 24, 31-155 Cracow, Poland

Joanna Kołodziej
Research and Academic Computer Network (NASK)
Kolska st 12, 01-045 Warsaw, Poland

KEYWORDS

Computational clouds; Cloud computing; Tasks scheduling; Energy savings; Cloud services modelling; Cloud security; Genetic algorithm.

ABSTRACT

Energy consumption is one of the most important problems in the era of Computational Clouds (CC). CC infrastructures must be elastic and scalable for being accessible by huge population of users in different geographical locations. It means also that CC energy utilization systems must be modern and dynamic in order to reduce the cost of using the cloud services and resources.

In this paper, we develop the novel energy saving strategies for resource allocation and task scheduling in computational clouds. We present the new energy–aware scheduling policies and methods of scaling the virtual resources. The idea of the proposed models is based on Dynamic Voltage and Frequency Scaling (DVFS) techniques of modulation of the power of microprocessors. Additionally, the proposed model enables the monitoring of the energy consumption, which is necessary for providing the scheduling under the security criterion. The efficiency of the proposed models has been justified in the simple empirical analysis. The obtained results show the need to maintain a balance between energy consumption and task schedule execution.

I. INTRUDUCTION

The main idea of the efficient resource and service providing in computational clouds is based on the virtualization of the available resources. Typical cloud cluster is a multi-tenant environment, in which many virtual machines (VMs) may be implemented at the same physical computational server. The efficiency and performance of such VMs depend on the on virtualization policy and characteristics of the hardware, i.e. virtual disk configuration and allocation policy, the speed of the physical processor, etc. [18], [15].

Considering security and privacy, cloud system should deliver appropriate security operations for each uploaded task [21] or should deliver the tools for the administrators for building the security infrastructures [22].

In this paper, we define the novel energy optimization strategies for task scheduling in the cloud environment. The optimization of the consumption of the energy in computational clouds is achieved through specialized load balancing methods and scaling of the VMs. Additionally, model is used for monitoring the utilization of the energy consumed for processing the security operations. Based on the monitoring results, the users may take their decisions on security parameters (security levels) and configuration of VMs. For instance, the users may generate the large (long) or small (short) keys for cryptographic procedures. Such key scaling services are available in Amazon Cloud, RackSpace, OpenStack and Google Clouds ([1], [5], [4],[3]).

The paper is organized as follows. In section II, we present the methods for modelling and calculation energy consumption for the activated Virtual Machines. In Section III, we specify security policies and parameters related to the Independent Batch Scheduling in computational clouds. Section IV describes the methods of calculation total energy consumption, and scaling of VMs. In section V, the scheduling problem is defined as multiobjective optimization task with energy consumption and security as the major scheduling criteria. In Section (VI), the scenarios of the energy management in CC are presented. The proposed methodology is empirically evaluated through simple numerical tests in Section VII. The paper is summarized in Section VIII.

II. VM POWER CONSUMPTION

There are many methods of measurement and optimization of power and energy consumed by the VMs in clouds. Power consumption of a physical server in the cloud infrastructure can be easily measured by using the well known methods designed for microprocessors [27], [28], [20]. This Problem is different in the case of VMs [11], [30]. The power of the physical server which is necessary for the utilization of the VM allocated at this server (we will call it " VM power" in the rest of this paper) cannot be estimated by using just the hardware methods. Some specified technologies, such as Watts UP PRO Power or APIs enable the measurement of the power consumption of CPU, memory, IO, devices, disks and networks (see CloudWatch metrics service for Amazon Cloud [2]). The amount of energy consumed by the VMs in computational cloud depends on many factors. Most of them results from the virtualization itself, the implementation of VMs at

Proceedings 31st European Conference on Modelling and Simulation ©ECMS Zita Zoltay Paprika, Péter Horák, Kata Váradi, Péter Tamás Zwierczyk, Ágnes Vidovics-Dancs, János Péter Rádics (Editors)
ISBN: 978-0-9932440-4-9/ ISBN: 978-0-9932440-5-6 (CD)

available servers and configuration of the physical infrastructure. Models of the energy utilization for VMs can be defined as modifications of the models for utilization of the physical resources, Such modifications are made by using the additional characteristics of the VMs architecture.

Let us denote by P_{Static} the power necessary for the preparation of the physical server for running the processes and implementation of the VMs (ready for work). Let $P_{Virtual}$ be the dynamic power consumed by VMs allocated at that server. The total power necessary for that physical server can be defined as follows [23]:

$$P_{Phys} = P_{Static} + \sum_{i=1,..,m} P(VM_i) = P_{Static} + P_{Virtual}, \quad (1)$$

where $P(VM_i)$ denotes the energy consumed by i-th instance of VM. Different methods are used to estimate this value. The non-observable variable $P(VM_i)$ is found based on observable P_{Phys}. Such methods are defined as relevant mathematical models with the most power-related resources as independent variables. The parameters of such models are estimated based on collected samples. The information may be collected by hyper-visor (black box method), or, in contrary, using white box method based on running proxy program on each VM [19].

In the model defined by Li, at al. in [31], the power consumption is calculated based on the utilization of CPU, operational memory and hard disk.

Bohra in [8] proposes a model that distinguish the baseline and active power consumption. In that paper, the independent variables are monitoring hardware performance of different Cloud system components and VMs energy consumption is estimated based on that measurements.

Krishnan in [30] used not only CPU utilization, but also memory consumption. Versick in [35] proposes a polynomial model. In this model network interface card (NIC) power consumption and hard disk power consumption is measured. The energy utilized by the VM depends on the above measurements.

Betran, at al. [7] used the linear model for the measurement of the VM power. The VM power consumption depends on 9 independent variables, such as activity of first level cache and number of accesses per cycle.

Most of the presented models are linear mathematical models with independent variables. Contrary, Gaussian Stochastic Mixture model is proposed in [12]. All of mentioned models are not good enough for the illustration of the realistic cloud virtual resource allocation and scheduling problems, [34].

There are some tools developed for the measurement of the virtual machine energy, such as FitGreen [13], Julemeter [26] or system proposed by Murwantara [32]. But those algorithms are not integrated with the cloud platforms. They need a special policy for the access to the cloud physical layer. Therefore, they may be implemented only from the cloud provider level or in the Infrastructure as a Service (IaaS) layer as a separate component.

Another approach is presented in [14], where the authors try to reduce energy consumption of applications running in cloud environments. The paper describes several deployment configurations based on queuing networks, and quantitative analysis for prediction of application performance and energy consumption.

Lot of attention was paid for the problem of the physical resources power consumption. D. Cerotti et al. [10] considered the issue of modelling power consumption in multicore CPUS with multithreading and frequency scaling. The authors present non-linear model for energy consumption that takes into account dynamic frequency scaling and Hyper-Threading, which have a significant impact on the model effectiveness.

This review shows that the Virtual Energy can be estimated. Additionally, the energy optimization methods presented so far are based on different methods comparing to those presented in this paper.

III. BATCH TASK SCHEDULING

In this paper, we considered the problem of Independent Batch Scheduling in computational clouds. We used Genetic Algorithm as the main mechanism for the cloud schedulers [29], [16], [17]. In this paper, we consider energy utilization as additional scheduling criterion. The main scheduling model is based on Expected Time to Compute (ETC) matrix, adopted to virtual machines (ETC_V). ETC_V matrix can be defined as follows:

$$ETC_V = [ETC_V[j][i]]_{j=1,\ldots,n;i=1,\ldots,m}, \quad (2)$$

where

$$ETC_V[j][i] = wl_j/cc_i, \quad (3)$$

in which cc_i is the computational capacity of i-th VM and wl_j is the workload of j-th task; n and m are respectively, number of tasks and number of VMs.

Based on the ETC_V matrix, we defined another $SBETC$ (Security Biased Expected Time to Compute) matrix which contains the additional security bias (SB) parameter b^i in order to reflect the security issues. Full description of this model can be found in [25].

The main objective of the scheduling is to find an optimal solution for specified criteria. The major objective for batch scheduling is the makespan that can be defined as follows:

$$C_{\max} = \min_{S \in Schedules} \left\{ \max_{j \in Tasks} C_j \right\}, \quad (4)$$

where C_j is the time when j-th task is finalized. $Tasks$ is the set of the tasks in the batch, and $Schedules$ is the set of all possible schedules, which can be generated for the tasks from that batch.

The batches of tasks are generated in non-deterministic time intervals. Therefore, each batch may have different number of tasks and the dimensions of the related SBETC matrices can be also various. The

detailed description of that process is presented in [24] and [25].

The measure that may be used for energy optimization is energy efficiency. It is calculated for the particular schedule s in the following way:

$$E_{efficiency}(VM_i) = \frac{\sum_{j=1,\ldots n,} wl_j \delta_{i,j}}{E(VM_i)} \quad (5)$$

where $\delta_{i,j}(s)$ is the binary factor. $\delta_{i,j}(s) = 0$ when the task number j is not scheduled for the machine i. $\delta_{i,j}(s) = 1$ when it is.

IV. ENERGY CALCULATION

A. Constant VM characteristics

Let E_{sec} denotes the energy necessary for running security operations. Then E_{total} is the total energy spend for particular batch of tasks. This energy is calculated for each schedule. In case of computational capacity of VM is constant and VMs are fully loaded during task running, only two energetic states are considered. They are: busy (100% computational power used for tasks calculations) and idle state. Let's assume that: t^i_{idle} - the time when i-th VM is idle; t^i_{busy} - the time when VM is calculating tasks; P^i_{idle} - the power necessary for VM to keep idle state; and P^i_{busy} - the power consumed by VM when is calculating tasks. The power necessary for security operations is assumed to be the same as in *busy* mode.

The above parameters are different for different schedules and can be defined as follows:

$$t^i_{busy} = max_{j \in Tasks\ scheduled\ \ for\ \ VM_i} C_j \quad (6)$$

$$t^i_{idle} = C_{max} - t^i_{busy} \quad (7)$$

$$t^i_{sec} = \sum_{j \in Tasks\ scheduled\ \ for\ \ VM_i} b^i_j \quad (8)$$

The overall energy may be expressed in the following way:

$$E_{total} = \sum_{i=1}^{m} \int_0^{C_{max}} Pow_{VM_i}(t)dt = \sum_{i=1}^{m} (P^i_{idle} * t^i_{idle} + P^i_{busy} * (t^i_{busy} + t_{sec})) = E_{task} + E_{sec} \quad (9)$$

B. VM scaling and re-provisioning

Very effective tool for saving energy in CC is VMs scaling. Scaling services allows to scale instances capacity up or down automatically or manually according to the user's needs. Cloud providers offers the following two main scaling methods:

- with re-provisioning: when scaling is done with change of allocation of virtual CPUs (vCPUs), memory, storage, or network resources. The scaled VM has to be shutdown (that lasts t_{close} sec.), new VM have to be configured and created (that lasts t_{open} sec.). It consumes relevant portion of system power.
- without re-provisioning, when VM is not reallocated, but only the computational power of it is changed. It is done by the change of capacity and redeploying VM from the template.

After each modification of the computational capacity parameter, the model (9) have to be updated. The computational capacity transition may be realized with or without re-provisioning of the VMs. In the second case, the time for the deactivation of a given VM and implementation of the new one and the related energy for that operations have to be considered. Also the dimensions of ETC_V and $SBETC$ matrices have to be changed.

There are two scenarios for VM scaling:

- scenario α: before calculating the schedule for the new batch, when all old tasks were executed and the system is idle, waiting for the next batch,. It may be done with re-provisioning ($\alpha 1$) or without ($\alpha 2$);
- scenario β: after calculating the schedule for the new batch, when workload is known to adapt to it. It may be executed with re-provisioning ($\beta 1$) or without ($\beta 2$).

Decision of the Cloud provider about choosing the number of VMs and their computational capacities for the next stage of system functioning may be based on different criteria and objectives. This problem is beyond the scope of this paper. The example of such decision process was presented in [36]. It was based on Stalkerberg games strategies. In this paper we assume that such decision was made earlier and the number of VMs and their computational capacities is set. Therefore the scheduling is based on:

- $\alpha 1$: when old VM has to be closed to open the new one

$$SBETC[j][i] = w_j/cc_i + b(sd_j, w_j, tl_i, cc_i) + t^i_{close} + t^i_{open}, \quad (10)$$

$$E^{\alpha 1}_{total} = E_{total} + \sum_{i=1}^{m} (P^i_{close} t^i_{close} + P^i_{open} t^i_{open}); \quad (11)$$

is the energy consumed;

- $\alpha 2$: when old VM do not have to be to be closed but it needs rescaling only:

$$SBETC[j][i] = w_j/cc_i + b^i_j + t^i_{scale} \quad (12)$$

$$E^{\alpha 2}_{total} = E_{total} + \sum_{i=1}^{m} P^i_{scale} t^i_{scale}; \quad (13)$$

- $\beta 1$: the scheduling is made for

$$SBETC[j][i] = w_j/cc_i + b^i_j + t^i_{close} + t^i_{open}, \quad (14)$$

but the tasks are done according to the new re-provisioned computational capacity $\overline{cc_i}$, therefore $E^{\beta 1}_{total}$ consists elements related to the new version of VM.

- $\beta 2$: the schedule is calculated for

$$SBETC[j][i] = w_j/cc_i + b^i_j + t^i_{scale}, \quad (15)$$

but the tasks are run according to the re-scaled computational capacity: $\overline{cc_i}$, therefore $E^{\beta 1}_{total}$ also includes values for re-scaled version of VM.

In all above cases, the number of tasks in the batch may be of the range $i = 1, 2, \ldots, \overline{m}$ and the number of VMs may be of the range $j = 1, 2, \ldots, \overline{n}$.

V. ENERGY AWARE SCHEDULING OBJECTIVES

The problem of finding the schedule that minimizes the makespan with constant computational capacities may be written in the form:

$$argmin_{s \in Schedules} \sum_{i=1,..,m} \sum_{j=1,...,n} (\frac{wl_j}{cc_i} + b^i)\delta_{i,j}(s) \tag{16}$$

The problem of finding the schedule that minimizes the total energy in that case may be formulated as:

$$argmin_{s \in Schedules} \sum_{i=1,...,m} (\sum_{j=1,\delta_{i,j}(s)=1}^{n} P^i_{busy}(\frac{wl_j}{cc_i} + b^i) + \sum_{j=1,\delta_{i,j}(s)=0}^{n} P^i_{idle} t^i_{idle}) \tag{17}$$

For cases α and β the proper equations are constructed in the similar way, taking into account relevant energy levels given in section IV.

The energy consumption may also be considered as a complementary scheduling criterion together with the makespan as the main objective.

We may also be interested in finding the rate of energy spend on the security operation to energy spend on bare task calculation: E_{task}/E_{sec}.

The usage of the SBECT matrix enables to test different energy savings strategies by lowering or rising the trust level of VM. Furthermore, the complex simulation may be performed before real cloud environment modifications.

VI. ENERGY SAVING SCENARIOS

A. Strategies for scheduler

We proposed four concurrent models for monitoring energy and makespan during the scheduling process. They reflect the importance of short time of tasks calculation and energy savings.

1. Makespan based scheduling and monitoring of the energy. In any case when two schedules have the same (or close) makespan, the scheduler chooses that one with smaller energy level. This case is suitable for the situation when the makespan is the priority. We want to save the energy not compromising the makespan.
2. Energy based scheduling and monitoring of the makespan. When the two schedules have the same (or close) energy level, the scheduler chooses that with smaller makespan. This case will be executed when we would like to calculate energy efficient schedules and we may afford to wait for our tasks longer.
3. Makespan based scheduling until the desired level is reached, then tasks are scheduled according to the energy objective. The search is performed only among the schedules that has the desired or smaller makespan.
4. Energy based scheduling until the desired level is reached, then tasks are scheduled according to the makespan objective. We are looking among the schedules that has the desired or smaller energy level.

The reference model is the makespan based scheduling only.

B. Strategies for VM scaling

Each VM may rated according to the energy efficiency, see eq. 5. This monitoring supports the decision making process about cancelling particular VM and replacing it with more effective one.

VM scaling may be done according to the $\alpha 1$, $\alpha 2$, $\beta 1$ or $\beta 1$ models.

For example, the new computational capacities after the schedule was calculated but before the tasks are executed, ($\beta 1$), may be found by solving the following problem:

$$C_{max} - t^i_{scale} - \overline{t}^i_{scale} = \frac{1}{\overline{cc}_i}(\sum_{j=1,\delta_{ij}(s)=1}^{n} wl_j + \overline{b}^i_j) + \overline{t}^i_{idle} \tag{18}$$

where the 'bar' values are computed for the new configuration of the environment. The new computational capacity for the machine is accepted if is it profitable according to the energy expenditure. The scheduling problem may be NP-complete due to many tasks [29]. The problem of finding the particular VM is not so time consuming. Cloud providers offers limited versions of instances.

VII. EVALUATION OF DEVELOPED MODELS

A. Tests of energy aware scheduling vs makespan scheduling

The aim of the test was to examine the makespan and energy consumption for the schedules that was calculated using different criteria.

The tests were evaluated using platform for simulation the Cloud environments called SimGrid [33].

Tested strategies are described in sec. VI-A Five types of VMs were assumed (see tab.I). The SURF component was used to simulate the execution of activities on resources [9]. For simulating the VM starting, running, scaling and closing the *pstates* were used.They allow to declare power states when the VM is switched off, the idle state power consumption and energy necessary for fully loaded VM. In case of Frequency scaling of the physical CPU of the VM the linear model in between fully load and idle state is assumed. The $watt^{state}_{per}$ function was used to measure the power consumption of the VM in each state. The simulator enables also to get current speed (in FLOPS) for each energetic state, total energy consumed so far (see tab. I). Additionally, the number of tasks to distribute, the computation size of each task, the size of the files associated to each task and a list of VMs that will accept

those tasks may be specified. The scheduling algorithm was implemented using C++. Optimization module for solving problems 16 and 17 was implemented in MATLAB programming environment.

TABLE I: Characteristics of VMs declared in SimGrid used for simulation

VM number	Speed GFLOPS	Energetic profile min:max in Watts
1	0.02	90:105
2	0.05	93:110
3	0.1	100:120
4	0.2	150:170
5	0.3	200:230

TABLE II: Measured energy consumption for benchmark task where $wl = 10$ GFLOPS and total simulation time is equal 510 seconds.

VM nr	After sleep for 10 sec. Joules	After exec. 10 Gflops Joules	Exec. time Seconds	Total energy Joules
1	900	53400	500	53400
2	930	22930	200	50830
3	1000	13000	100	53000
4	1500	10000	50	77500
5	2000	9666.66	33.34	103000

The computational capacities of VMs were measured by the benchmark test (see tab. II). They were computed using execution time of benchmark task having workload of 10 GFLOPS and was stated to be equal to the declared VMs speed (see tab. III). The P^i_{idle} and P^i_{busy} per second were calculated using energy consumed in benchmark execution time. Idle machine was simulated using sleep mode.

TABLE III: Calculated VMs characteristics

cc_i	P^i_{idle}	P^i_{busy}	E_{eff}	P^i_{open}	P^i_{close}	P^i_{scale}
0.02	90	106.8	0.09	63	27	54
0.05	93	114.6	0.08	65	28	56
0.1	100	130	0.07	70	30	60
0.2	150	200	0.05	105	45	90
0.3	200	290	0.03	140	60	120

The time of VM opening was assumed to be the same for all VMs: $t^{scale} = t_{close} = t_{open} = 1sec$. Additionally, the environment was modelled so that $P^i_{open} = 70\% P^i_{idle}$, $P^i_{close} = 30\% P^i_{idle}$, $P^i_{scale} = 60\% P^i_{idle}$.

Fist test considered very simple VM loading of 5 tasks scheduled for 5 machines. Testing workloads were: $wl_1 = 1, wl_2 = wl_3 = wl_4 = 2, wl_5 = 8GFLOPS$. Moreover, tasks having the same workload are indistinguishable and may be computed interchangeably. For such a case direct search of best schedules was implemented. Scheduling according to the makespan objective (see model 1) resulted in the makespan=50 sec. For that schedule, energy consumed by VMs was 258004 Watts. The resulted mapping of tasks for consecutive VMs (1 to 5, according to the table I) was: 1, 2, 2, 2, 8. This mapping was denoted as schedule 1. Scheduling according to the energy objective (see model 2) resulted in the schedule with energy consumption equal 257596 Watts. This schedule makespan was 160 sec. The resulted mapping of tasks for consecutive VMS was: 1, 8, 2, 2, 2. They formulated schedule 2.

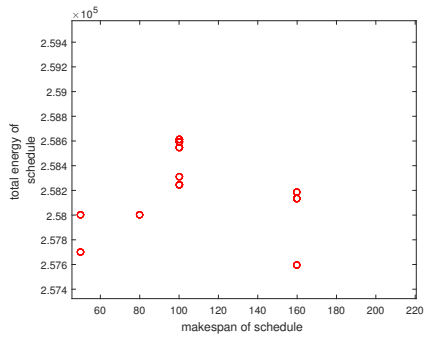

Fig. 1: Energy consumed for different schedules

Fig. 1 shows that for the same makespan there is possible to find schedules consuming less energy. Conversely, for the same energy there are several schedules having more or less beneficial makespan. Therefore finding suboptimal (or optimal) solutions according to the models 3 and 4 is possible and may be profitable.

B. Tests of strategies for VM scaling

The aim of these tests was to examine the influence of proper re-scaling of VMs on the energy consumption. Tested strategies are described in sec. VI-B. For these chosen schedules presented in the previous subsection, the re-scaling strategy $\beta 2$ was incorporated. For schedule 1. only the first VM was busy for the whole makespan time, see tab. IV. Therefore MVs 2, 3 and 4 was examined for the possibility off re-scaling, see tab. V. The most beneficial re-scaling was: VM 3 scaled VM 2, VM 4 scaled VM 2 and VM 5 scaled VM 4. It enabled to save energy, keeping makespan of batch at the same level. For schedule 2. second machine was not considered for re-scaling because is not idle. First VM machine was also excluded, because there is no VM with smaller energy consumption that VM number 1. The most beneficial re-scaling was: VM 3 scaled VM 2, VM 4 scaled VM 2 and VM 5 scaled VM 1.The example result of scaling in tern of idle time of the machine and best schedule 2. is the following:

- before re-scaling the VM 5 was busy for 6.6 sec consuming 290 Watt/sec. and was idle for 200 sec. consuming 200 Watts/sec.
- after re-scaling the VM 5 into VM 4, it was busy for 10 sec consuming 200 Watt/sec. and was idle for 149 sec. consuming 150 Watts/sec., additionally 1 sec. and 120 Watt was consumed into re-scaling operation

Incorporating scaling scenarios when schedule is ready is very beneficial. The scaled VMs will do the

assigned work in the same makespan, but the energy consumption is much lower.

TABLE IV: VMs loading for the chosen optimal schedules

Tasks	Idle time for consecutive VMs	Working time for consecutive VMs
Schedule 1. 1, 2, 2, 2, 8	0,10,30,40,23.4	50,40,20,10,26.6
Schedule 2. 1, 8, 2, 2, 2	110,0,140,150,153.4	50,160,20,10,6.6

TABLE V: VMs power consumption and time of tasks running time, before and after scaling for generated best suboptimal schedules: best schedule 1: 1,2,2,2,8 and best schedule 2: 1,8,2,2,2. Only the beneficial scaling possibilities are presented.

Schedule 1 old VM → new VM	Energy before scaling	Energy after scaling
3 → 2	5600	5481
4 → 3	8000	5590
4 → 2	8000	5511
5 → 4	12394	9470
Schedule 2		
3 → 2	16600	15711
3 → 1	16600	16050
4 → 3	24500	16590
4 → 2	24500	15741
4 → 1	24500	16040
5 → 4	32594	24470
5 → 3	32594	23570
5 → 2	32594	15747
5 → 1	32594	12510

C. *Tests of energy aware scheduler with advanced workload*

Scheduling many tasks require using advanced method for searching the optimal schedule. We implemented Genetic Algorithm (GA) for finding the schedules given by strategies presented in sec. VI. Four strategies for scheduler are described in sec VI-A.

The models were implemented in C++. Due to the fact that makespan and energy values are *double* type, the 3% difference between values was chosen as the increment distinguishing one type of *double* real value from the second one.

The model was tested for set of 20 VMs, see tab. VI. These characteristics were obtained by test on real infrastructure (see [6]). Each batch consists 200 tasks. Every run of the GA was tested for 20 set of parameters configurations. The single gene represents the schedule. In our approach, the final result is the whole population that represents the suboptimal schedule (see: 2. Crossover operation means swapping tasks among virtual resources from respectively, set with best and worse fitted individuals. Mutation operation was not used. During each epoch the population is changed, but the size of population remains the same.

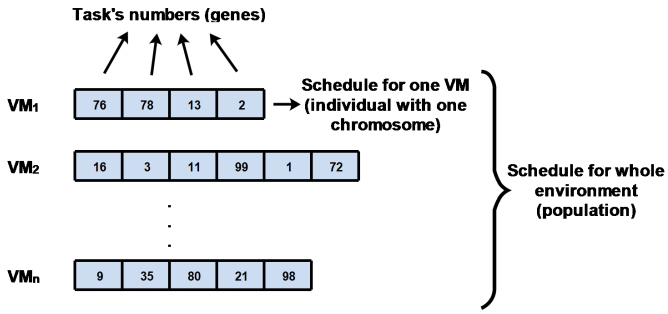

Fig. 2: Mapping tasks into genes

TABLE VI: Tested VMs

i	cc_i	P^i_{idle}	P^i_{busy}	i	cc_i	P^i_{idle}	P^i_{busy}
1	0.2	71.9	57.52	2	0.7	75.4	60.32
3	1.0	74.5	59.6	4	1.9	82.0	65.6
5	1.7	78.6	62.88	6	2.4	71.0	56.8
7	10.6	73.0	58.4	8	62.9	128.7	102.96
9	71.8	124.7	99.76	10	50.4	122.5	98.0
11	57.8	123.6	98.88	12	0.4	55.9	44.72
13	2.2	60.1	48.08	14	2.7	60.4	48.32
15	4.2	62.7	50.16	16	4.3	62.5	50.0
17	9.8	60.6	48.48	18	47.7	64.9	51.9
19	1.71	17.1	13.68	20	1.73	17.4	13.92

After some iterations, average individuals are frozen and tasks are assigned to the recourses permanently (see: fig. 3). Therefore the number of swapped tasks is decreasing. The best and worst individuals are chosen to be crossovered. During each epoch the new population is created and evaluated. The fitness function for the GA was assumed to be equal the makespan or total energy consumed per schedule.

Exchanging two tasks according the makespan influences the energy consumption. This is due to the fact that makespan depends only on the tasks scheduled on the machine that needs the longest tasks to complete the work. Energy consumption differs for the schedules having the same makespan, but different task distribu-

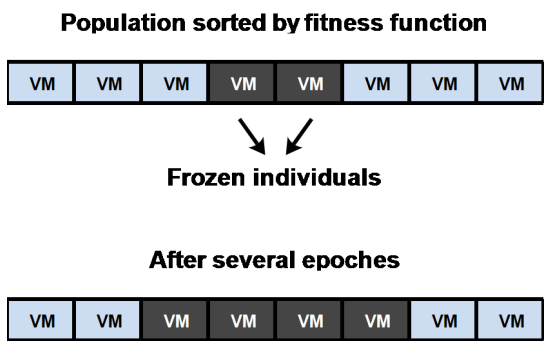

Fig. 3: Excluding average individuals from exchanging tasks (freezing)

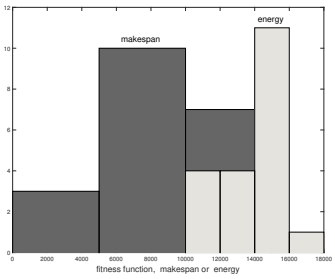

Fig. 4: Histogram of generated fitness function for scheduling according to the makespan and total energy

tion. It depends on the working (busy) and idle time of all the recourses in the system. Therefore, larger number of crossovers modifies the value of total energy fitness GA function. GA was also more sensitive for scheduling according to the energy objective. Fig. 4 presents the value of fitness function for 20 GA runs after 2000 epochs. Incorporating energy objective after initial stage of makespan scheduling (model 3) results in both smaller mean makespan, and lowering energy consumption of the system, comparing to the scheduling based on energy criterion only (model 2). The results of testing energy saving scenarios 1 - 4 are presented in the tab. VII.

The best result for the proposed GA was obtained where scheduling was done according to the energy criterion first, and makespan criterion as second (model 4).

TABLE VII: Mean values of makespan and total energy for 20 initial populations of GA running, tested 4 scenarios (see sec. VI-A)

Model	Mean makespan	Mean total energy
1	8732.2 sec.	86149.3 W
2	13844.3 sec.	136546 W
3	13337.4 sec.	118166 W
4	7345.5 sec.	76902.7 W

The experimental results show, that the crossover for scheduling according to the energy fitness function is less effective. In this scenario we are exchanging tasks between worst and best individuals. There is a need for formulation another criterion for finding better populations offspring.

VIII. SUMMARY

In this paper we developed and implemented a new model of energy and security aware Independent Batch Scheduler. We defined four scenarios for monitoring of the energy utilization and makespan during scheduling process. We also developed four models for scaling VMs in order reduce the energy consumption. Additionally, we presented the short overview of methods for estimation energy consumption in virtualized environments, and existing energy scaling methods for VMs.

The experimental results presented in the paper demonstrate and confirm the effectiveness of proposed models. The best result for scheduling of advanced workload was obtained for the last model, where the major scheduling objective was energy and the second makespan.

In the future, we would like to avoid the work on the cloud simulators and implement our models on the realistic CC platforms working with the OpenStack standards.

ACKNOWLEDGEMENT

This article is based upon work from COST Action IC1406 "High-Performance Modelling and Simulation for Big Data Applications" (cHiPSet), supported by COST (European Cooperation in Science and Technology)

References

[1] Amazon Cloud Scaling Service.
[2] Amazon CloudWatch.
[3] Google Cloud Scaling Service.
[4] OpenStack Cloud Scaling Service.
[5] Rackspace Cloud Scaling Service.
[6] P. Benner, P. Ezzatti, E. Quintana-Ortí, and A. Remón. *On the Impact of Optimization on the Time-Power-Energy Balance of Dense Linear Algebra Factorizations*, pages 3–10. Springer International Publishing, 2013.
[7] R. Bertran, Y. Becerra, D. Carrera, V. Beltran, M. Gonzalez, X. Martorell, J. Torres, and E. Ayguade. Accurate energy accounting for shared virtualized environments using pmc-based power modeling techniques. In *2010 11th IEEE/ACM International Conference on Grid Computing*, pages 1–8, Oct 2010.
[8] A. E. H. Bohra and V. Chaudhary. Vmeter: Power modelling for virtualized clouds. In *2010 IEEE International Symposium on Parallel Distributed Processing, Workshops and Phd Forum (IPDPSW)*, pages 1–8, April 2010.
[9] H. Casanova, A. Giersch, A. Legrand, M. Quinson, and F. Suter. Versatile, scalable, and accurate simulation of distributed applications and platforms. *Journal of Parallel and Distributed Computing*, 74(10):2899 – 2917, 2014.
[10] D. Cerotti, M. Gribaudo, P. Piazzolla, R. Pinciroli, and G. Serazzi. *Modeling Power Consumption in Multicore CPUs with Multithreading andFrequency Scaling*, pages 81–90. Springer International Publishing, Cham, 2016.
[11] M. Colmant, M. Kurpicz, P. Felber, L. Huertas, R. Rouvoy, and A. Sobe. Process-level power estimation in vm-based systems. In *Proceedings of the Tenth European Conference on Computer Systems*, EuroSys '15, pages 14:1–14:14, New York, NY, USA, 2015. ACM.
[12] G. Dhiman, K. Mihic, and T. Rosing. A system for online power prediction in virtualized environments using gaussian mixture models. In *Design Automation Conference*, pages 807–812, June 2010.
[13] C. Dupont, T. Schulze, G. Giuliani, A. Somov, and F. Hermenier. An energy aware framework for virtual machine placement in cloud federated data centres. In *2012 Third International Conference on Future Systems: Where Energy, Computing and Communication Meet (e-Energy)*, pages 1–10, May 2012.
[14] M. Gribaudo, T. T. N. Ho, B. Pernici, and G. Serazzi. *Analysis of the Influence of Application Deployment on Energy Consumption*, pages 87–101. Springer International Publishing, 2015.
[15] D. Grzonka. The Analysis of OpenStack Cloud Computing Platform: Features and Performance. *Journal of Telecommunications and Information Technology*, 3:52–57, 2015.
[16] D. Grzonka, J. Kołodziej, and J. Tao. Using artificial neural network for monitoring and supporting the grid scheduler performance. In *28th European Conference on Modelling and Simulation, ECMS 2014, Brescia, Italy, May 27-30, 2014*, pages 515–522, 2014.
[17] D. Grzonka, J. Kołodziej, J. Tao, and S. U. Khan. Artificial neural network support to monitoring of the evolution-

ary driven security aware scheduling in computational distributed environments. *Future Generation Computer Systems*, 51:72–86, 2015.

[18] D. Grzonka, M. Szczygiel, A. Bernasiewicz, A. Wilczynski, and M. Liszka. Short analysis of implementation and resource utilization for the openstack cloud computing platform. In *29th European Conference on Modelling and Simulation, ECMS 2015, Albena (Varna), Bulgaria, May 26-29, 2015. Proceedings.*, pages 608–614, 2015.

[19] C. Gu, H. Huang, and X. Jia. Power metering for virtual machine in cloud computing-challenges and opportunities. *IEEE Access*, 2:1106–1116, 2014.

[20] T. W. Harton, C. Walker, and M. O'Sullivan. Towards power consumption modeling for servers at scale. In *2015 IEEE/ACM 8th International Conference on Utility and Cloud Computing (UCC)*, pages 315–321, Dec 2015.

[21] A. Jakobik. *Big Data Security*, pages 241–261. Springer International Publishing, Cham, 2016.

[22] A. Jakobik. A cloud-aided group rsa scheme in java 8 environment and openstack software. *Journal of Telecommunications and Information Technology : JTIT*, (2):53–59, 2016.

[23] A. Jakóbik, D. Grzonka, J. Kołodziej, A. E. Chis, and H. Gonzalez-Velez. Energy Efficient Scheduling Methods for Computational Grids and Clouds. *Journal of Telecommunications and Information Technology*, 1, 2017.

[24] A. Jakobik, D. Grzonka, J. Kolodziej, and H. González-Vélez. Towards secure non-deterministic meta-scheduling for clouds. In *30th European Conference on Modelling and Simulation, ECMS 2016, Regensburg, Germany, May 31 - June 3, 2016, Proceedings.*, pages 596–602, 2016.

[25] A. Jakobik, D. Grzonka, and F. Palmieri. Non-deterministic security driven meta scheduler for distributed cloud organizations. *Simulation Modelling Practice and Theory*, in press. (available online 4 November 2016).

[26] A. Kansal, F. Zhao, J. Liu, N. Kothari, and A. A. Bhattacharya. Virtual machine power metering and provisioning. In *Proceedings of the 1st ACM Symposium on Cloud Computing*, SoCC '10, pages 39–50, New York, NY, USA, 2010. ACM.

[27] H. Kataoka, D. Duolikun, T. Enokido, and M. Takizawa. Power consumption and computation models of a server with a multi-core cpu and experiments. In *2015 IEEE 29th International Conference on Advanced Information Networking and Applications Workshops*, pages 217–222, March 2015.

[28] H. Kataoka, A. Sawada, D. Duolikun, T. Enokido, and M. Takizawa. Energy-aware server selection algorithms in a scalable cluster. In *2016 IEEE 30th International Conference on Advanced Information Networking and Applications (AINA)*, pages 565–572, March 2016.

[29] J. Kołodziej. *Evolutionary Hierarchical Multi-Criteria Metaheuristics for Scheduling in Large-Scale Grid Systems.* Springer Publishing Company, Incorporated, 2012.

[30] B. Krishnan, H. Amur, A. Gavrilovska, and K. Schwan. Vm power metering: Feasibility and challenges. *SIGMETRICS Perform. Eval. Rev.*, 38(3):56–60, Jan. 2010.

[31] Y. Li, Y. Wang, B. Yin, and L. Guan. An online power metering model for cloud environment. In *2012 IEEE 11th International Symposium on Network Computing and Applications*, pages 175–180, 2012.

[32] I. M. Murwantara and B. Bordbar. A simplified method of measurement of energy consumption in cloud and virtualized environment. In *Proceedings of the 2014 IEEE Fourth International Conference on Big Data and Cloud Computing*, BDCLOUD '14, pages 654–661, Washington, DC, USA, 2014. IEEE Computer Society.

[33] M. Quinson. Simgrid: A generic framework for large-scale distributed experiments. In *2009 IEEE Ninth International Conference on Peer-to-Peer Computing*, pages 95–96, Sept 2009.

[34] J. Read. What is an ECU? CPU Benchmarking in the Cloud.

[35] I. Wassmann, D. Versick, and D. Tavangarian. Energy consumption estimation of virtual machines. In *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, SAC '13, pages 1151–1156, New York, NY, USA, 2013. ACM.

[36] A. Wilczyński and A. Jakóbik. Using Polymatrix Extensive Stackelberg Games in Security–Aware Resource Allocation and Task Scheduling in Computational Clouds. *Journal of Telecommunications and Information Technology*, 1, 2017.

AUTHOR BIOGRAPHIES

AGNIESZKA JAKÓBIK (KROK) received her M.Sc. in the field of stochastic processes at the Jagiellonian University, Cracow, Poland and Ph.D. degree in the field of neural networks at Tadeusz Kosciuszko Cracow University of Technology, Poland, in 2003 and 2007, respectively. From 2009 she is an Assistant Professor. Her e-mail address is: agneskrok@gmail.com

DANIEL GRZONKA received his B.Sc. and M.Sc. degrees with distinctions in Computer Science at Cracow University of Technology, Poland, in 2012 and 2013, respectively. Currently, he is Research and Teaching Assistant at Cracow University of Technology and Ph.D. student at Jagiellonian University in cooperation with Polish Academy of Sciences. He is also a member of Polish Information Processing Society and IPC member of several international conferences. His e-mail address is: grzonka.daniel@gmail.com. For more information please visit: www.grzonka.eu

JOANNA KOŁODZIEJ is an associate professor in Research and Academic Computer Network (NASK) Institute and Department of Computer Science of Cracow University of Technology. She is a vice Head of the Department for Sciences and Development. She serves also as the President of the Polish Chapter of IEEE Computational Intelligence Society. She is also a Honorary Chair of the HiPMoS track of ECMS. Her e-mail address is: joanna.kolodziej68@gmail.com. The detailed information is available at www.joannakolodziej.org