Journal of Electrical Engineering and Information Technologies, Vol. **4,** No. 1–2, pp. 29–40 (2019)
Article 159 ISSN 2545–4250
Received: April 24, 2019 e: ISSN 2545–4269
Accepted: May 13, 2019 UDC: 004.738.5:004.383.2.052

Original scientific paper

ENERGY EFFICIENT AND OPTIMAL ALLOCATION OF RESOURCES IN CLOUD COMPUTING NETWORKS

Maja Lazarevska, Toni Janevski

Faculty of Electrical Engineering and Information Technologies,
"Ss. Cyril and Methodius" University in Skopje,
Rugjer Bošković bb, P.O. box 574, 1001 Skopje, North Macedonia
*majalazarevska92@gmail.com // tonij@feit.ukim.edu.mk

A b s t r a c t: Cloud computing is an emerging technology designed as replacement of the traditional physical hardware solutions. Consisted of virtual resources provided on user demand, it allows scalable pay-as-you-go model of data sharing. Cloud computing is a model that allows the division of a shared network with commonly configured computing resources where, before being assigned to customers, the available resources must be reviewed and scheduled using an efficient resource scheduler. Most existing standard schedulers do not meet the corresponding standards and user requirements. Therefore, additional research and analysis are needed in order to create a scheduler that would perform an energy efficient and optimal allocation of resources in any cloud computing network. In this research work, a new task allocation and scheduling algorithm is proposed, based on the use of newly created metric, enabling the selection of the most suitable server, frame and module in a standard cloud computing architecture.

Key words: cloud computing; green cloud simulator; scheduling algorithms; task allocation

ЕНЕРГЕТСКИ ЕФИКАСНА И ОПТИМАЛНА АЛОКАЦИЈА НА РЕСУРСИ ВО CLOUD COMPUTING МРЕЖИ

А п с т р а к т: Cloud computing е нова технологија дизајнирана како замена за традиционалните физички хардверски решенија. Составен од виртуелни ресурси доставени по барање на корисникот, моделот дозволува споделување на податоците да биде со скалесто плаќање на услугите (pay-as-you-go). Cloud computing претставува модел кој дозволува поделба на заедничка мрежа со заеднички конфигурирани компјутерски ресурси каде што достапните ресурси, пред да им бидат доделени на корисниците, мора да бидат прегледани и распоредени користејќи ефикасен распоредувач на ресурси. Повеќето постојни стандардни распоредувачи не ги исполнуваат соодветните стандарди и кориснички барања. Поради таа причина, потребни се дополнителни истражувања и анализи со цел да се создаде распоредувач кој би овозможил енергетско ефикасна и оптимална алокација на ресурси во која било cloud computing мрежа. Во овој истражувачки труд се предлага нов алгоритам за алокација и распределба на задачи и ресурси, базиран на користењето на новосоздадената метрика, овозможувајќи избор на најсоодветен сервер, рамка и модул во една стандардна cloud computing архитектура.

Клучни зборови: cloud computing; green cloud симулатор; распоредувачки алгоритми; алокација на ресурси

INTRODUCTION

Cloud computing applies to the on-demand delivery of IT resources and applications over the Internet with a so-called pay-as-you-go payment method. A cloud computing network is consisted of number of data centers distributed to different locations in the world that contain a large number of server groups. These infrastructures must be constantly maintained by the service providers, taking into account not only the work infrastructure, but also the energy and environmental issues. In order to achieve better and more efficient energy performance, cloud services providers rely on scheduling algorithms, which aim to manage data centers through optimization and resource allocation. The purpose of the scheduling algorithms is to find and allocate resources for a particular task of the cloud system while meeting the user requirements and optimizing a particular function that takes into account

the satisfaction of the cloud user and also the cloud provider.

Problems with the performance of the scheduling algorithms include time to perform tasks, resource use, and energy consumption. The ideal resource scheduler would use less resources and less time to perform more tasks. Using a smaller number of resources is very important point, because it would mean lower power consumption, as well as the ability to create large and highly-performing cloud computing networks. On the other hand, it is not favorable to have inactive (idle) resources and jobs waiting in a buffer. Having an efficient scheduler of tasks becomes an urgent need with the increased use of modern computer systems and the need for optimal performance. Scheduler algorithms are responsible for mapping tasks submitted to the cloud environment in free resources, so that the total response time and latency are minimized, while the capacity and resource utilization are maximized [1].

Even though some of the most common conventional task scheduling algorithms and their extensions have achieved very good results for different types of computer systems, there are still problems like the long waiting time, the lack of resources to perform the tasks, leaving unfinished tasks or high energy consumption. Due to this reason, it is necessary to create new algorithms for allocating tasks that would solve these problems and would allow a scheduler that would require less waiting time, less time for allocation of resources, lower energy consumption and the possibility of a successful allocation of resources without having unsuccessful or unfinished tasks.

This paper proposes a new scheduling algorithm that uses combination of server, rack and module selection functions in a single uniform metric.

RELATED WORK

Resource allocation and virtualization

Some of the most important challenges facing today's cloud computing implementation are energy efficient resource allocation and the choice of the most appropriate and optimal server for delivering resources without enormous power consumption. Servers represent the main consumers of electricity in cloud data centers, due to:

- *Poor use of the server* – As the size of the data centers increases, the number of servers continues to grow, which leaves the existing data centers unused, while the newly added servers are being completely exploited [2].
- *No power usage* – The servers remain inactive and do not process information 85–95% of the time [3], but the existence of inactive servers leads to consumption of 70% of the power even in cases when they are not used [4].
- *Lack of standardized metrics for server energy efficiency* – In order to ensure optimization of energy efficiency and selection of the most suitable recourses, it is necessary to use energy efficiency metrics on servers.

Scheduling algorithms

The allocation of resources and tasks consists of identifying, selection, allocating resources to each incoming user request so that all the user requirements specified in that request would be met, as well as the specific requirements of the cloud provider and finally submission of tasks for each specific resource. In a cloud-computing environment, the allocation of resources is managed by the cloud providers through virtualized technology.

Not all existing scheduling algorithms are suitable for real-time tasks in an imprecise cloud environment, since their approach refers to the assumption that the cloud environments are deterministic with computerized decisions statically assigned during the scheduling process.

Round Robin (RR) algorithm for allocating resources provides equal allocation of each task per server (uniformity). This simple algorithm allows balancing the load, minimization of unsuccessful and unfinished tasks, while the network overload, congestion and delay can be totally avoided [5]. But, since no server is set up as idle or off and all servers are active during all the task allocation process, this algorithm might not be very energy efficient.

Random scheduler allows random allocation of resources on the available server (a uniform default decision). This leads to long waiting time for serving the tasks and due to that, unsuccessful or unfinished tasks. This algorithm is not complex because it does not require any preconditions or preprocessing, but on the other hand does not allow optimal and energy-efficient server selection.

Green scheduler is allocating tasks in an energy conscious way. It assigns tasks to a minimum number of computer servers, which means the allocation of tasks is performed by leaving the largest

number of inactive servers compared to other schedulers. After servers are unused and inactive for a long time, they automatically would turn off [6]. The scheduler monitors the use of network switches on the path in a continuous way and it collects information about the current load of resource providers in order to select the first resource provider that will successfully perform the task. Whenever there is congestion, the scheduler stays away from the overcrowded routes, even though they might lead to servers that can meet the requirements. In this way, the number of unfinished and unsuccessful tasks, as well as the average response time, are minimized. On the other hand, the number of scheduling routes outperforms the capabilities of the switches and it can cause network overload.

Other researched method is the DENS methodology (Data center Energy-efficient Network-aware Scheduling), which combines energy-efficient resource allocation with knowledge of the network architecture. The proposed approach provides a compromise between the work consolidation (minimizing the number of used servers) and the distribution of traffic patterns (to avoid problems in the data center network) [7].

eSTAB (Energy-Efficient Scheduling for Cloud Computing Applications with Traffic Load Balancing) is another scheduler implemented in the GreenCloud simulation environment [8]. This scheduler includes the traffic requirements of the cloud applications allowing efficient allocation of tasks and balancing the traffic load in data center networks. Moreover, the communication requirements are as important as the computing requirements. eSTAB consists of two main steps: firstly, selecting a group of servers in which at least one of them can meet the computing requirements of the tasks, and then from the selected group of servers, choosing a server with the smallest computational capacity, but enough to meet the requirements.

HEROS (Energy-Efficient Load Balancing for Heterogeneous Data Centers) represents a method that works at the rack level and its decision function is based on the use of performance metrics per Wat of servers using network connections. The final decision is obtained as a multiplication of the server selection function and the function of the communication potential. This energy efficient scheduler has two contradictory goals: power consumption and average response time (flowtime) [9].

DCEERS (Data Center-wide Energy-Efficient Resource Scheduling framework) [10] is an algorithm that assigns a minimum number of resources to tasks by calculating the minimum cost of distribution using the Bender decomposition theory.

PROPOSED MODEL

Power model

The proposed solution for resource allocation in the distributed computer architecture is designed to optimize the consumption of energy in cloud computing data centers and to provide the optimal choice of server. In order to better understand and determine consumption, power models are used as prototype of the virtual implementation of scheduling algorithms. The proposed solution uses the linear power model based on three-level data center architecture, which consists of an access layer (composed of computer servers-hosts), aggregation layer (responsible for routing) and central layer (allows connection between several different modules), as presented in Figure 1.

Energy management techniques

The proposed solution uses a combination of two energy management techniques: Dynamic Voltage and Frequency Scaling (DVFS) and Dynamic Power Management (DPM).

DPM provides the largest amount of energy savings through disconnection of devices (including all device components), but if it becomes necessary to re-enable the computer servers, a considerable amount of energy would be used for it. For a more effective DPM methodology, the scheduler must have a minimal set of computer resources to maximize the concurrent servers that can be shut down (or get into sleep mode). Given that many servers in data centers are inactive most of the time, they can be turned off or set in sleep mode during periods of time when they are not used and then engage when needed. This energy management scheme works well in systems that consist of a homogeneous computer server, while in heterogeneous environments it does not show significant results [11].

DVFS technology, on the other side, adjusts the hardware power consumption according to the applied computer load. It performs dynamic scaling of the voltage and frequency of the processor during the execution of tasks [12]. Although this method aims to reduce power consumption, it only applies to server level and it remains insufficient to fully optimize the power consumption.

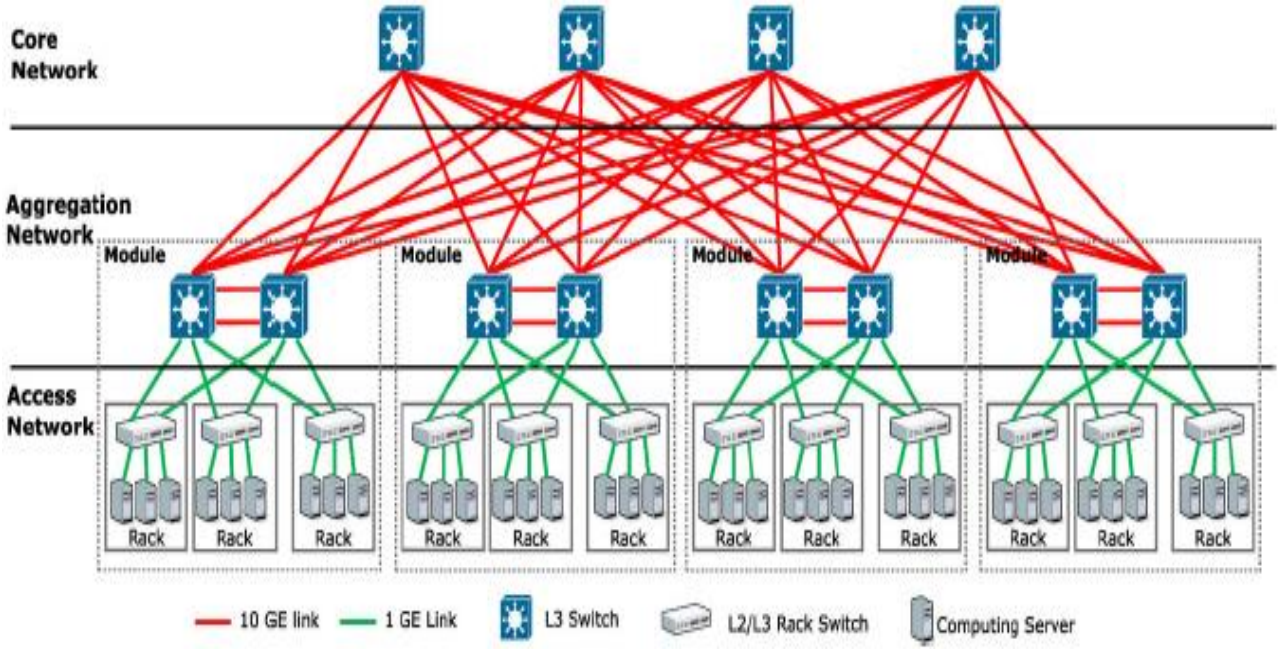

Fig. 1. Three-tier data center architecture [11]

Scheduling algorithm function

Unlike traditional resource allocation solutions that focus on modelling data centers composed of homogeneous computer servers, the proposed solution works with data centers composed of homogeneous and heterogeneous computer servers through a hierarchical model of three-level architectture of a date center (server, rack, module). There are several types of heterogeneous computer servers: commodity, high-performance (HPC) and low-performance micro (micro) servers. Commodity servers are the most inefficient servers that use the concave power function, high-performance servers use the convex power feature and are the most efficient servers, while low-performance micro servers use the linear power function and are the servers that will be used in this proposed solution.

The following formal model explains the heterogeneity: If we have access to $n + 1$ computers, a server C_0 and a cluster (group of computers) C consisted of n computers, C_1, ... C_n, that operate at different speeds and have uniform distribution of the work, each cluster C_i will run one unit of work in ρ_i time units. The vector $< \rho_1, ... \rho_n >$ represents the heterogeneous profile of the cluster. Moreover, the existence of a heterogeneous environment brings additional challenges in the decision-making process for the allocation of resources [9].

The new metric M is the main decisive function in the newly proposed scheduling algorithm as shown in Equation (1), which is a combination of the server level Fs, the rack level Fr and the module level Fm and their coefficients of importance a, b and c.

$$M = a \cdot Fs + b \cdot Fr + c \cdot Fm \qquad (1)$$

The server that has the greatest value for the metric M, which takes into account the whole architecture and resource allocation path, is the server that should perform the task. The coefficients of importance a, b and c define the importance of the main components of the three-level architecture in the metric M. Moreover they define which layer of the cloud computing architecture would have priority when considering the whole cloud architectture. They are experimentally defined in order to provide the highest energy consumption gain according to the characteristics of each component and its importance in the proposed algorithm, the level of importance of each layer of the cloud architecture as well as the value of the sum of the three coefficients of importance ($a + b + c = 1$). Given these parameters and characteristics, the values for a, b and c are set to 0.5, 0.3 and 0.2, respectively. Some variations of these parameters might add gains in other parameters, while add degradations in others. For the purposes of this study, the experimentally defined coeficients are the most suitable for optimal energy consumption in such three level cloud architecture.

In cases when the values of the metric M are the same for two or more servers, then according to the priority given by the coefficient of importance,

firstly the server level *Fs* is being compared and the server with the highest server level would perform the task. Likewise, if the values of the server levels of two or more servers are the same, the rack level *Fr* is being compared and, finally, if the rack levels are identical, then the module level *Fm* is being compared. Additionally, if all the architecture levels are the same for one or more servers, then the server with the fastest response time is selected to perform the task.

- **Server**

The function for selecting a suitable server is represented by Equation (2) as a combination of the performance per Watt per server (PpW), server load function and communication potential [13]:

$$E_S(J,u) = P_pW_S(l) \cdot L_S(I) \cdot Q(u). \quad (2)$$

The performance per Watt (PpW) as shown in Equation (3) is used to emphasize the energy efficiency and the energy performance of the server:.

$$PpW_S(l)=Perf_S(l)/P_S(l). \quad (3)$$

$Perf_s(l)$ represents a function that defines the server performance as a load function, while $P_s(l)$ represents the energy consumption function, with different definition for different energy management methods (DPM or DVFS), as shown in Equation (4) and Equation (5), respectively.

$$P_s(l) = P_{idle} + \frac{P_{peak} - P_{idle}}{2}(1 + l - e^{\frac{l}{\tau}}) \quad (4)$$

$$P_s(l) = P_{idle} + \frac{P_{peak} - P_{idle}}{2}(1 + l^3 - e^{-(\frac{l}{\tau})^3}). \quad (5)$$

$P_s(l)$ gives the energy consumption per server depending on the used energy management method where the load level l is in the interval [0,1], P_{peak} is the server's energy consumption when there is maximum load, P_{idie} is the energy consumption of the server when there is a minimum load and τ is the coefficient of scaling in the perimeter [0.5, 0.8] and gives the level of use of the server when consuming energy. One disadvantage of the load-dependent metric is the fact that servers are most effective when fully loaded, but also it leads to overload and drastically reduce performance and energy efficiency [9].

The second perimeter that the server selection function depends on is the server load function, which similarly to the DENS algorithm [7], represents an amount of two sigmoid functions, as preented in Equation (6). The first sigmoid function gives the shape of the main sigmoid, while the latter refers to maximizing the value of the load on the server. The constant ε defines the slope size. This function considers the high values of the load and the impact they have on the efficiency.

$$L_s(l) = \frac{1}{1 + e^{-10(l-\frac{1}{2})}} - \frac{1}{1 + e^{-\frac{10}{\varepsilon}(l-(1-\frac{\varepsilon}{2}))}} \quad (6)$$

The third parameter associated with the server selection function refers to the communication potential for equal redistribution of resources, based on the HEROS algorithm, as defined in Equation (7):

$$Q(u) = e^{-(\frac{2u}{U_{max}})^2}. \quad (7)$$

Where u represents the current link load, while U_{max} is the maximum load on the link.

- **Rack**

Using similar principle as the selection of most suitable server, the rack selection function $F_r(l, u)$, as shown in Equation (8), depends on the load function of the rack $L_r(l)$, the function for communication potential for equal redistribution of resources $Q(u)$ and the performance per Watt for the corresponding rack $P_pW_r(l)$.

$$F_r(l,u) = PpW_r(l) \cdot L_r(l) \cdot Q(u). \quad (8)$$

The performance per Watt per frame is obtained as the sum of the individual performance per Watt values for each server in the rack (Equation (9)), while the load function of the frame $L_r(l)$ represents a sum of the individual load of the servers in the rack (Equation (10)), and the constant n represents the number of servers in the rack. The communication potential has unique value for the whole architecture, as given in Equation (7).

$$PpW_r(l) = \sum_{i=1}^{n} PpW_s(l) \quad (9)$$

$$L_r(l) = \frac{1}{n}\sum_{i=1}^{n} L_s(l) \quad (10)$$

- **Module**

Similar to the servers and the racks, the modules are selected using the proposed function in Equation (11), depending on the performance per Watt for the module $P_pW_m(I)$, the load function of

the module $L_m(I)$ and the communication potential for equal redistribution of resources $Q(u)$:

$$F_m(I,u) = PpW_m(I) \cdot L_m(I) \cdot Q(u) \quad (11)$$

In this case, the parameter PpW of the module is obtained as the sum of the values of the parameter for each server in the module (Equation (12)). The load function of the module $L_m(I)$ represents the sum of the individual loads on the servers in the module (Equation (13)), while the constant k represents the number of racks in the module.

$$PpW_m(I) = \sum_{i=1}^{k} PpW_r(I) \quad (12)$$

$$L_m(I) = \frac{1}{k}\sum_{i=1}^{k} L_s(I) \quad (13)$$

Algorithm 1 shows the pseudo-code of the proposed scheduler for energy efficient resource allocation and task scheduling.

Algorithm 1 Algorithm for energy efficient and reliable resource allocation and task scheduling

```
procedure RESOURCE ALLOCATION ALGORITHM(Fs, Fr, Fm)
Fs: Function for server selection
Fr: Function for rack selection
Fm: Function for module selecton

    while Tasks are received in data centers do
        Power consumption calculation (DPM/DVFS power management methods);
        Performance per Watt calculation;
        # PpW= Perfs/Ps
        Load calculation;
        Comunication potental calculation;
        Check server availability;
        Choosing the most energy efficient server;
        # Fs=PpWs x Ls x Q
        Check rack availability;
        Best parent selection;
        Choosing the most energy efficient rack;
        # Fr=PpWr x Lr x Q
        Check module availability;
        Choosing the most energy efficient module;
        # Fm=PpWm x Lm x Q
        Calculating metrics M;
        # M=aFs + bFr + cFm
        if Metrics M1 > Metrics 2 then
            Server 1, rack 1 and module 1 perform task;
        else Metrics M1 < Metrics 2
            Server 2, rack 2 and module 2 perform task;
        end if
        Task allocation finished;
    end while
end procedure
```

IMPLEMENTATION

The proposed algorithm is implemented in the GreenCloud simulator specified for simulations and analysis of cloud computing networks. The Green Cloud simulator is an extension of the Ns_2 network simulator. It provides detailed models of consumed energy by the elements of data centers (servers, switches and links). It also provides simulations of communications in the data center architecture at the packet level. The Figure 2 shows the structure of the GreenCloud extension mapped into three-level architecture of data centers, used for the proposed solution and simulation scenarios [11].

At the access level are the computer servers which are responsible for carrying out the tasks. The choice of an appropriate and optimal server for allocating and performing tasks is essential for the proper functionality of the cloud computing networks. In GreenCloud, the server components implement single nodes that have a limit of processing power in millions of instructions per second or FLOPS (floating point operations per second), associated memory size / storage resources, and contain different task allocation algorithms. As shown in the picture, servers are grouped into racks that connect via the Top-of-Rack (ToR) switches to the access part of the network. In general, the GreenCloud simulator uses two power models, a linear model and a low-power model. For the implemented solution, the linear power model is used.

The power model of the server components is proportional with the state of the server and its use of the CPU. The state of the server is important, because even an inactive server in the network consumes about 2/3 of the energy compared to the fully loaded configuration. This consumption of the inactive server is in order to allow operation of the memory, disks and I/O resources in the current configuration. For the remaining 1/3, power consumption increases linearly with increasing of CPU levels.

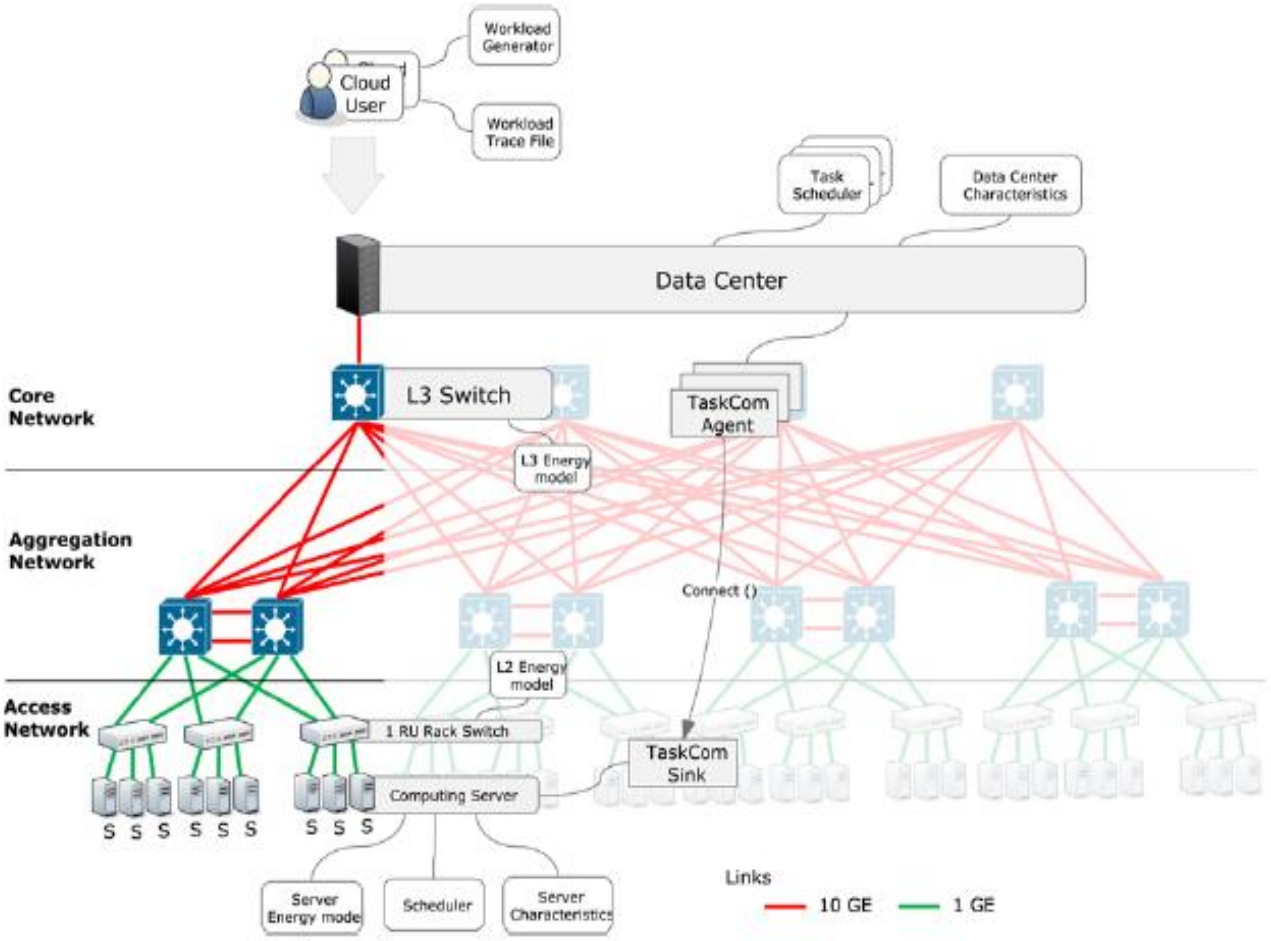

Fig. 2. Implementation of the three-tier data center architecture in the GreenCloud simulator [11]

SIMULATION

Cloud simulation is needed to analyze and test cloud systems in order to reduce complexity and increase service quality [14], [15].

The used cloud simulator is the GreenCloud Simulator, which is a sophisticated open source simulator with user friendly GUI on a package level that allows display of communication, power, physical and power saving models. Full implementation of the TCP / IP protocol is supported, but the simulation time is slightly longer than the other commonly used cloud computing simulators. One of the main advantages of the GreenCloud simulator is the detailed modelling of the energy consumed by the data centers and by each of the elements of the data centers. It is used to develop new technologies and solutions in terms of monitoring, resource allocation, distribution of work, virtualization and migration of virtual machines, as well as optimization of communication protocols and network infrastructures. About 80% of the GreenCloud code is implemented in C++, and the remaining 20% are in the form of Tool Command Language (TCL) scripts.

The configuration parameters of the four scenarios for evaluation of the capabilities of the newly introduced scheduler are shown in Table 1. One of the advantages of the NEW scheduler is the ability to test every server that is in the rack and thus the chances of reducing energy consumption in homogeneous and heterogeneous configurations are much greater. A homogeneous environment has the same type of servers in the configuration. A heterogeneous environment has different types of servers, so the total number of servers might be deployed linearly or exponentially. In the researched scenario, the *tcl* simulation script for homogeneous configuration uses only commodity servers, while for the heterogeneous scenario three types of servers are used, such as: commodity, HPC (high-power computing) and micro servers. The main differences between the types of servers are: the number of core cores, the MIPS per core, the total MIPS, the minimum power, the maximum power, the existence of

a hard drive. These parameters are essential for the choice of the metrics allowing optimal task allocation and lower energy consumption. Additionally, the larger number of different server types might influence lower or higher energy consumption (for example HPC servers need higher power, whereas micro servers need lower power).

Table 1

Data centers and topology parameters

	Configuration parameters	Homogeneous network	Heterogeneous network
Data centers and Topology	Core switches	2	2
	Aggregation switches	4	4
	Access switches	5	5
	Servers in rack	20	20
	Total number of racks	10	10
	Total number of servers	200	200
	Server type 1 (commodity)	200	66
	Server type 2 (HPC)	0	30
	Server type 3 (micro)	0	104
	Core links	10 Gb/s, 3.3 μs	100 Gb/s, 3.3 μs
	Aggregation links	10 Gb/s, 3.3 μs	10 Gb/s, 3.3 μs
	Access links	1 Gb/s, 3.3 μs	1 Gb/s, 3.3 μs
	Average number of delivered tasks	46045/138135	35976 / 117158
	Average number of delivered tasks per server	230.2 / 690.7	179.88 / 585.8
	Task scheduling mechanism in servers	DVFS/DNS	
	Task scheduling mechanism in switches	DVFS	
	Number of cloud users	1	

RESULTS

To understand and demonstrate the contribution of the research and the efficiency of the new resource scheduler, set of parameters are evaluated using existing scheduling algorithms as references, such as: Round Robin, Random and Green, which are implemented in the GreenCloud simulator by default [16]. The analysis of the obtained results was carried out according to the following evaluation parameters:

- *Total energy* – Total energy consumption (sum of energy consumption of servers, primary, aggregation and access switches).
- *Energy on servers* – Consumption of server energy.
- *Energy (core / aggregation / access) switching* – Energy consumption of basic / aggregation / access switches.
- *Average response time* – The time difference between the time the query was running and the time when the tasks were allocated to the servers available.
- *Failed tasks* – Tasks detected that cannot be performed before the execution deadline and are abandoned by the servers.
- *Unfinished tasks* – Tasks that do not leave the data center at all (the sum of unsuccessful tasks and tasks that do not establish communication before the end of the simulation).

Four different scenarios that change the heterogeneity (heterogeneous and homogeneous environment) and the server load (30% and 80%) are simulated and evaluated. The performed simulations include the practical results given by the NEW scheduler and also the simulated results from the standard defined simulators which correspond to the practical results given from the references. Finally the schedulers in all four scenarios are compared and a corresponding conclusion is given.

The first scenario (Scenario 1) refers to a small homogeneous three-level configuration with server load of 30% consisting of more than 46 thousand tasks, or about 230 tasks per server. Considering the results for total energy consumption, it can be noticed that the Green scheduler and the New scheduler provide significantly better results than the Round Robin and Random scheduling algorithms. The reason for this is that neither Round Robin nor Random allow the existence of a "sleep" mode of the servers. Despite the favorable results for total energy consumption and energy consumption per server, the unfinished tasks in the Green scheduler represent 10% of the total number of tasks assigned to the system, and also the average response time is the longest compared to other schedulers. For these reasons, these results cannot be accepted as successful. Accordingly, it can be concluded that the performances of the new scheduler exceed Green, Round Robin and Random scheduling algorithms.

The second scenario (Scenario 2) consists of a small heterogeneous three-level configuration, with server load of 30%. The heterogeneous configuration, by definition, deploys a smaller number of tasks, so appropriate and less energy consumption would be needed for a successful distribution of energy. Round Robin and Random schedulers are not designed to support heterogeneous configurations, as concluded by the higher values for total energy consumption, unsuccessful and unfinished tasks (about 50% of the total number of tasks delivered) compared to the NEW and Green schedulers. On the other side, both New and Green scheduler have no unsuccessful or unfinished tasks. Regarding the average response time, it can be noted that all four types of scheduling algorithms produce similar results that do not affect the selection of the most favorable scheduling algorithm. The new scheduler gives better results for the total power consumption while the Green scheduler gives better results for the average response time. The results are shown in Fig. **3**, Fig. **4**, Fig. **5** and Fig. **6**.

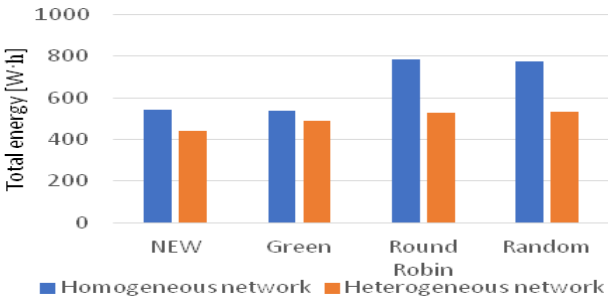

Fig. 3. Total energy when servers load = 0.3

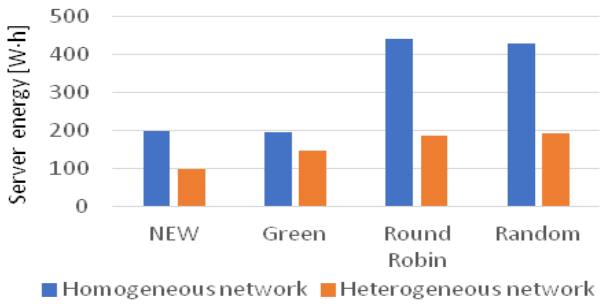

Fig. 4. Energy of servers when servers load = 0.3

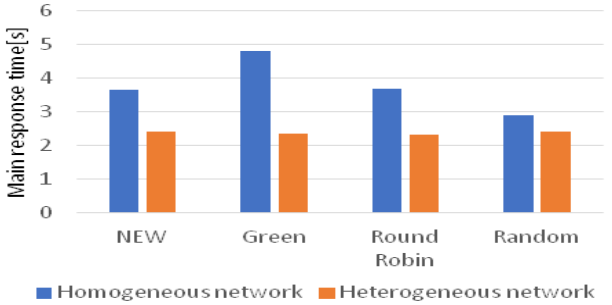

Fig. 5. Main response time when servers load = 0.3

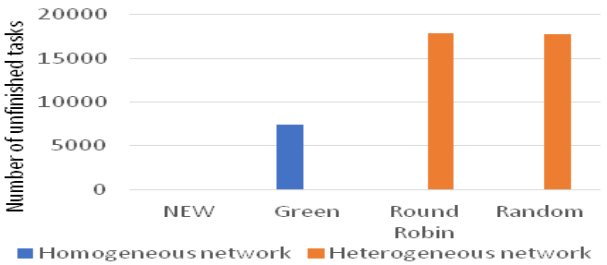

Fig. 6. Unfinished tasks when server load = 0.3

The third scenario (Scenario 3) refers to a homogeneous environment, with a data centers load of 80%. According to the simulated results, RR and Random schedulers provide lower values for total energy consumption in comparison with NEW and Green schedulers. Also, the average response time for Round Robin and Random is less than that of NEW and Green, due to the lower complexity of the schedulers. Additionally, the homogeneous environment does not have unfinished tasks.

The fourth simulated scenario is Scenario 4, which represents a small heterogeneous three-level configuration that has a load of 80%. The total power and energy on the servers of the New and Green scheduler is higher due to increased server load and due to the complexity of algorithms based on optimal server selection through a combination of optimal rack and module selection. RR and Random give large values for unfinished and unsuccessful tasks that make them unstable and unacceptable for heterogeneous environments. Concerning the average response time, it can be noticed that the Green and New schedulers have a higher value than RR and Random. The results are presented in Fig. **7**, Fig. **8**, Fig. **9** and Fig. **10**.

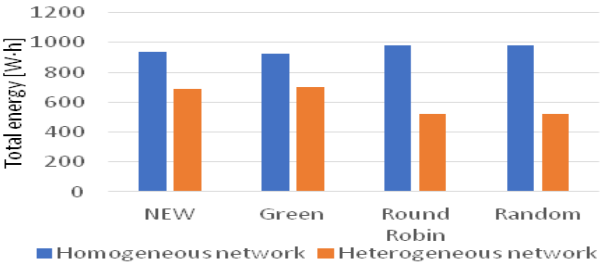

Fig. 7. Total energy when servers load = 0.8

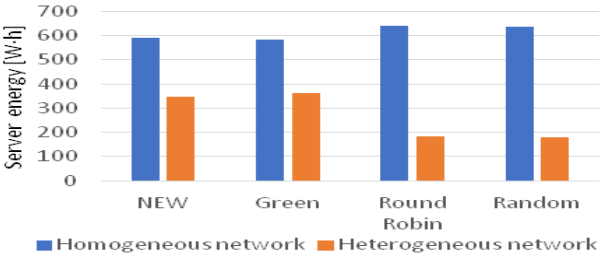

Fig. 8. Energy of servers when servers load = 0.8

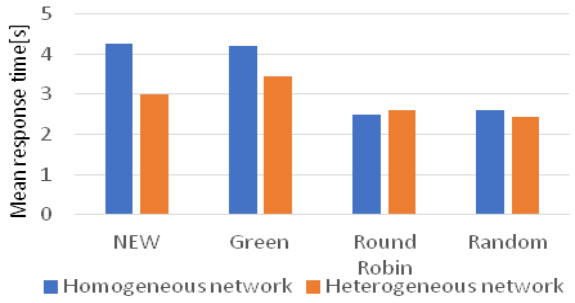

Fig. 9. Mean response time when servers load = 0.8

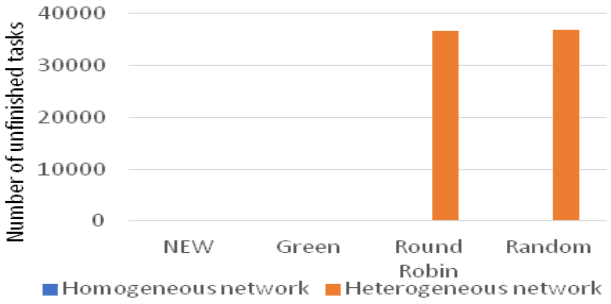

Fig. 10. Unfinished tasks when servers load = 0.8

Additionally, since in the reviewed scenarios the heterogeneous network is consisted of large number of micro servers that have lower values for minimum and minimum power usage, it is normal that the value of the total energy in heterogeneous environment (in any scheduler) is lower than in homogeneous environments. However the results show that generally the NEW scheduler gives better results regarding the energy optimization in comparison to the standard schedulers [17]. With this, it can be concluded that the NEW scheduler is correctly designed for heterogeneous and homogeneous configurations regardless of the load value.

Using the tracking records for each simulation, as shown in Fig. **11** and Fig**. 12**, the consumed energy per server as well as the number of tasks assigned per server can be displayed and analyzed. Since the Green scheduler uses only part of the allocation servers, the energy consumption focuses mainly on the active servers, while the power consumption of inactive servers is lower. On the other side, the NEW scheduler uses almost all servers to allocate tasks and leaves a very small number of inactive servers. Accordingly, the energy consumption is a sum of the energy consumed on each server. Additionally, the existence of active and inactive servers means that the total number of received tasks is distributed only to the active servers. That means that the number of tasks performed depends on the proportion of power consumption.

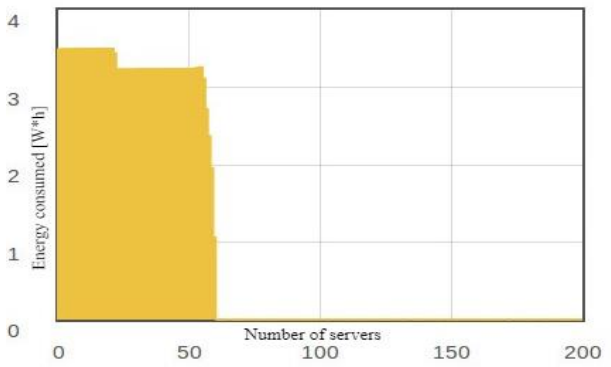

Fig. 11. Energy consumed per server – Green scheduler

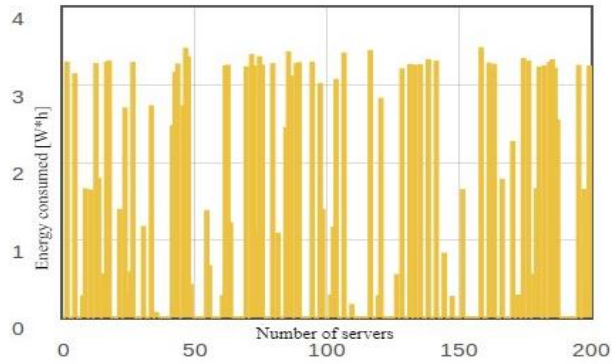

Fig. 12. Energy consumed per server – New scheduler

CONCLUSION

The new scheduling algorithm is based on a special metric that takes into account the full architecture of the cloud computing network, as well as the heterogeneity. It depends on the load function of the components, the communication potential for equal redistribution of resources and the metrics of efficiency per Watt.

All described simulations are performed on a Virtual Machine with Ubuntu 12.04 installed operating system, which enables the use of the cloud simulator GreenCloud. Using different analyzes the NEW algorithm proved its energy efficiency and optimality in different simulation scenarios compared to several reference algorithms.

Table 2 provides an overview of the final results for the NEW algorithm. It gives the total energy consumption, servers energy and average response time for the NEW algorithm for each of the four simulated scenarios. Having greater server load means that the total energy consumption is mainly dependent on the server's energy consumption, and the power consumption of the switches is lower than that the consumption of servers. Additionally, the

values of the total energy are lower in a heterogeneous environment than in a homogeneous environment for each of the schedulers. This is due to the use of three types of servers in the heterogeneous environment and with that less allocation tasks.

T a b l e 2

Overview of the final results of the NEW scheduler

Scenario	Total energy	Server energy	Average response time
	W·h	W·h	s
Homogeneous, load = 0.3	541.7	199.3 (37%)	3,64
Heterogeneous, load = 0.3	441	98.6 (22.3%)	2,39
Homogeneous, load = 0.8	935.2	592.8 (63.3 %)	4,25
Heterogeneous, load = 0.8	690.9	348.5 (50%)	3,01

To conclude, the NEW scheduler simplifies the complexity of a heterogeneous system, because it examines the overall architecture of the system, composed of different types of servers, racks, modules, links and different types of switches, when deciding on the resource allocation. It also normalizes the power and capacity functions and allows different simulation parameters to be used. As a result, it improves the quality of service for cloud computing applications and allows lower energy consumption as well as successful distribution of incoming tasks, with a slight degradation of the average response time due to the complexity of the technique used in the algorithm.

The future work of this research would focus on performing additional and more complex simulations with different parameters and scenarios (multi user). It is expected that the newly proposed algorithm would be applicable in different scenarios with different heterogeneity, different load and number of users. Also further research on resource allocation methods are needed in order to provide a higher percentage of profit in terms of energy efficiency and the number of completed and successful tasks. It is also expected that the proposed resource allocator algorithm will be implemented in the real cloud computing network in the future. The real cloud computing environment consists of a growing number of cloud users and different number and characteristics of user requirements, as well as more complex requirements for resources, which can lead to further challenges. In addition, in order to improve the performance of the task scheduler, it is necessary to reconsider the proposed model and allocation algorithm, such as possible change of the main metric for selecting the server, rack and module, the calculation of the functions (communication potential and load function), as well as the modification of the coefficients of importance (a, b, c).

REFERENCES

[1] Srivastava, P., Khan, R.: A Review on Cloud Computing, *International Journals of Advanced Research in Computer Science and Software Engineering*, ISSN: 2277–128X, Vol. **8**, Iss. 6, pp 17–20 (June 2018).

[2] Jadeja, Y., Modi, K.: Cloud computing-concepts, architecttures and challenges, *2012 International Conference on Computing, Electronics and Electrical Technologies* (ICCEET), 2012.

[3] Baun, C., Kunze, M., Nimis, J. Tai,: *Cloud Computing – Web-Based Dynamic IT Services*, Springer Verlag, Berlin–Heidelberg, 2011.

[4] Jararweh, Y., Jarrah, M., Kharbutli, M., Alshara, Z., Alsaleh, M. N., Al-Ayyoub, M.: CloudExp: A comprehensive cloud computing experimental, *Simulation Modelling Practice and Theory*, Vol. **49**, pp. 180–192 (2014).

[5] Buttazzo, G. C.: Predictable scheduling algorithms and applications, In: *Real-Time Computing Systems*, Pisa, Springer, 2016.

[6] Monteiro, R. C., Dantas, M. A. R., Rodriguez, M. V. R. Y.: *Green Cloud Computing: An Experimental Validation*, High Performance Computing Symposium 2013 (HPCS 2013), 2014.

[7] Kliazovich, D., Bouvry, P., Khan. S. U.: DENS: data center energy-efficient network-aware scheduling, *2010 IEEE/ACM Int'l Conference on Green Computing and Communications & Int'l Conference on Cyber, Physical and Social Computing*, December 2010.

[8] Kliazovich, D., Arzo S. T., Granelli, F., Bouvry, P., Khan, S. U.: eSTAB: Energy-Efficient Scheduling for Cloud Computing Applications with Traffic Load Balancing, *2013 IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing*, December 2013.

[9] Guzek, M., Kliazovich, D., Bouvry, P.: HEROS: Energy-Efficient Load Balancing for Heterogeneous Data Centers, *IEEE 8th International Conference on Cloud Computing*, 2015.

[10] Shuja, J., Bilal, K., Madani, S., Khan, S.: Data center energy efficient resource scheduling, *Cluster Computing*, Vol. **17**, number. 4, pp. 1265–1277 (2014).

[11] Kliazovich, D., Bouvry, P., Audzevich, Y., Khan, S. U.: GreenCloud: A Packet-Level Simulator of Energy-Aware Cloud Computing Data Centers, *2010 IEEE Global Telecommunications Conference GLOBECOM 2010*, pp. 1–5, December 2010.

[12] Zhang, Q., Zhani, M. F., Zhang, S., Zhu, Q., Boutaba, R. Hellerstein, J. L.: Dynamic energy-aware capacity provisioning for cloud computing environments, *9th International Conference on Autonomic Computing, ICAC '12*, New York, NY, USA, 2012.

[13] Fiandrino, C., Kliazovich, D., Bouvry, P., Zomaya, A. Y.: Performance and Energy Efficiency Metrics for Communication Systems of Cloud Computing Data Centers, *IEEE Transactions on Cloud Computing*, Vol. **5**, Nº. 4, December 2017.

[14] Daylami, N.: The Origin and Construct of Cloud Computing, *International Journal of the Academic Business World*, Vol. **9**, No. 2 (2015).

[15] Zehe, D., Cai, W., Knoll, A., Aydt. H.: Tutorial on a modeling and simulation cloud service, *2015 Winter Simulation Conference (WSC '15)*, Piscataway, NJ, USA, 2015.

[16] Atiewi, S., Yussof, S., Ezanee, M.: A Comparative Analysis of Task Scheduling Algorithms of Virtual Machines in Cloud Environment, *Journal of Computer Science*, Vol. **11**, no. 6 (2015).

[17] Lago, D., Madeira, E., Medhi, D.: Energy-Aware Virtual Machine Scheduling on Heterogeneous Bandwidths' Data Centers, *Journal IEEE Transactions on Parallel and Distributed Systems*, Vol. **7** (2017).