2018 26th Mediterranean Conference on Control and Automation (MED)
Zadar, Croatia, June 19-22, 2018

Data Center Server Energy Consumption Optimization Algorithm*

Iulia Stamatescu[1], Stéphane Ploix[2], Ioana Făgărăşan[1] and Grigore Stamatescu[1]

Abstract— **Understanding, modelling and reducing data center power consumption is a complicated undertaking. Individual data centers are highly complex systems, with a number of interacting mechanical, electrical and computational subsystems. The paper presents an optimization algorithm which reduces the energy consumed by the informational systems by assign virtual machines to physical servers. This action is achieved taking into account the different degrees of freedom of the used resources. The considered problem is formulated as a mixed integer linear programming problem where all transfer functions, constraint functions and objective function are expressed under a linear form.**

I. INTRODUCTION

Due to the rapid growth of computing power and with their consolidation in larger facilities the data center consumed energy has reached a point of interest in the years 2000. If the topic of minimizing the consumed energy was originally a point of interest only for managers, suddenly researchers and regulators where interested as well. The response of the vendors and of the several associations was the publication of best practices for minimizing the energy consumption and promoting more efficient technologies.

Reducing the consumption of energy in a data center can be engaged at multiple levels as:

- In data center design and optimization tools helps to find an optimal sizing of the equipment [1], [2].
- Regarding the cooling of physical servers, many studies have been done in order to define an optimal equipment positioning for servers and storage arrays. [3], [4] and [5] propose to vary the speed of cooling fans to reduce operating costs in the period known as ”hollow”. More recently, the use of free cooling [6] is considered: it is used under external natural cooling conditions.

In terms of physical hardware, several improvements on the servers and storage arrays have seen the light of day.

A. Hardware level

After the reduction of the size of the engraving, the advent of micro-processors and multi-cores [7], [8] have increased very significantly the computers efficiency, sometimes expressed as Watt/MHz/cores. This technology is more present at that physical level of commercial servers rather the personal computers. Commercial servers often have around 16-32 cores [7], [9]. Another technology allows the reduction of the power consumption even more: the dynamic adjustment of voltage and frequency of cores in microprocessors [10].

B. Software level

Some strategies are already managing, in a global manner, the data center input and output data. These strategies have to reduce energy consumption while meeting the requirements bounded by a contract between the data center manager an his client: the Service Level Agreement (SLA) [11]. The most promising solution today is probably the virtualization which improves the energy efficiency without reducing the availability of the service. Nevertheless, static allocation of virtual machines on physical servers induces the making of significant safety margins to ensure the meeting of the commitments caught in the SLA. These margins nevertheless induce additional energy costs [12], [13]. The dynamic allocation, the number of requests, the availability requirement and the implicit time-varying of demands, are a new source of gain. Different algorithms are proposed as:

- An approach based on the FFD (First Fit Decreasing) algorithm, which is applied for positioning online virtual machines [14].
- An approach based on h1 filter [15] which isn't implemented.
- An approach based on ”limited look-a-head control” [16], which is a multi-objective approach that can take into account the nonlinear constraints. The approach is based on a ”predictive control model” in terms of application.
- An approach based on linear programming for dynamic positioning loads [17], [18].
- Another approach [12] presents a semi-static management. Starting from a learning algorithm, this approach aims to predict from a cluster of servers under requested servers. Taking into account the predictions result, the algorithm sorts the servers into three categories: servers underutilized, servers medium used and servers normally used. Taking into account this classification, the algorithm seeks to place for workloads of the under requested servers on the medium ones, keeping in mind the criterion for compliance with the SLA. If all the conditions are satisfied the migration becomes effective and energy can be saved. Nevertheless, this approach is questionable because the choice of servers that has to be turned off is done by iteration.

*The publication of this work is supported by University Politehnica of Bucharest, through the ”Excellence Research Grants” Program, UPB - GEX 2017. Identifier: UPB - GEX2017, Ctr. No. 59/2017 (CYNERGY).

[1] Researcheris with Automatic Control and Informatics Department, University POLITEHNICA of Bucharest, 313 Splaiul Independentei, 060042, Romania iulia.stamatescu@aii.pub.ro

[2] Researcheris with the G-SCOP Laboratory, INP Grenoble 46 Avenue Felix Viallet, 38031 Grenoble, France

- A second approach developed at IBM Lab [9] consists of processing data concerning applications which needs to be hosted in order to find the best possible cohabitation. A first algorithm "Correlation Based Placement" seeks, starting from predictions of load applications, an anti-correlation peak of each application to compose groups called "cluster" of applications that can be placed on servers sufficiently capable. These servers are presented in order for privileging energy efficiency. Another algorithm "Clustering Based Investment " aims to do an encapsulation of the applications under two levels (peak, normal). The applications are selected from a probability of occurrence. The following step is to group these envelops with a tolerance depending on the value of the peak and its time of occurrence and the normal level. Subsequently, a combination between difference envelops are used to build the bindings on existing servers in the data center.

The problem that this paper aims to solve is finding a power efficient configuration and a corresponding mapping algorithm for the multiple running applications. The major problem that dynamic migration has to deal with is to decide what is the best location for executing a new job, depending on the resources it requires in order to fulfil its QoS (quality of service) and according with the provider targets for power eefficiency, reliability, etc. The aim of the research is to find an algorithmic solution that allows to assign virtual servers to physical servers in order to minimize the overall energy consumption. This action will be achieved by taking into account the different degrees of freedom, the used resources, the physical server characteristics, the different levels of reliability and the global approach of the data center in order to fulfill the quality of services.

II. APPLICATION CHARACTERIZATION

It can be stated that each application is characterized by workload (U = requests/hour) and can de written as a function which depends on the server performance expressed by the CPU occupation according to application workload: $U_{server} = f(requests/hour)$. An application is considered to be composed of multiple services (an e-commerce web includes many different services: email, chat, searching, buying, reservation etc). For facilitating the problem, in the present paper, it will be considered that each application represents a service.

Application performance depends both on the type of application (games, computing application, graphic application, etc.) and on the server (hardware and software) where is hosted. Given n application and m server, there is $n \times m$ performance models $\mathcal{F}_{i/j}$ corresponding to $n \times m$ combinations of placement $< Application,\ Server >$ as showed in the matrix:

$$\begin{array}{c|cccc} & Server_1 & Server_2 & ... & Server_m \\ \hline Application_1 & \mathcal{F}_{1/1} & \mathcal{F}_{1/2} & & \mathcal{F}_{1/m} \\ Application_2 & \mathcal{F}_{2/1} & \mathcal{F}_{2/2} & & \mathcal{F}_{2/m} \\ ... & & & & \\ Application_n & \mathcal{F}_{n/1} & \mathcal{F}_{n/2} & & \mathcal{F}_{n/m} \end{array}$$

III. OPTIMIZATION STRATEGY

The proposed optimization strategy is based on the behaviors of elements included within data centers, modeled as transfer functions.

A. Formulation of the optimization problem as a mixed integer linear programming problem

The main idea of the scheduling policy is to migrate the virtual machines (*VM's*) in the data center servers to the available machines taking into account the different degrees of freedom that can be encountered. The scheduling police is turning off idle machines to minimize the global energy consumption. So in this scheduling policy, the considered freedom degrees are:

- The position of *VM* on server
- The *VM* state
- The Server State

The scheduling policy which uses consolidation and turning off idle machines is formulated as a mixed integer linear programming problem which must take into account the transfer functions:

- the model of each *VM* on each server represented at time i.
- the energy consumption model of server represented at time i and other constraints [19].
- the hosting capacity of physical machine which depends on the hardware and software requirements. It may be defined as a hosting server list predefined by the user and the maximum number of *VM* on each server.
- the virtualization overheads: *VM* creation, migration, shutting down of *VM*.

The degrees of freedom

Based on this information, the global optimization problem may be formulated as follows. The degrees of freedom are the number of independent pieces of information that can be found into an equation.

The first freedom degree is the placement of $VM's$ on servers. This can be is expressed as a hosting matrix. Given n applications A_i which is need to be placed on m physical servers S_j, a hosting matrix is formed by n rows and m columns. It is assumed that an application isn't scalable (it cannot be assigned to different virtual servers) and each application A_i runs on an assigned virtual machine VM_i. The problem is to find the best combinations $< VM_i,\ Server >$ during the anticipative duration T. Each cell in the hosting matrix is a variable $\mathcal{P}_{i/j}(t)$ with with $\mathcal{P}_{i/j}(t) \in \{0, 1\}$, which shows whether the VM_i is placed on the server S_j at the time t.

$$\begin{cases} P_{i/j}(t) = 1 & if\ VM_i \text{ is placed on server } S_j \\ P_{i/j}(t) = 0 & otherwise. \end{cases} \quad (1)$$

The set of variables $(\mathcal{P}_{i/1}(t), ..., \mathcal{P}_{i/m}(t))$ shows the degrees of freedom concerning the position of VM_i.

$$\begin{array}{c|cccc} & S_1 & S_2 & ... & S_m \\ \hline VM_1 & \mathcal{P}_{1/1}(t) & \mathcal{P}_{1/2}(t) & & \mathcal{P}_{1/m}(t) \\ VM_2 & \mathcal{P}_{2/1}(t) & \mathcal{P}_{2/2}(t) & & \mathcal{P}_{2/m}(t) \\ ... & & & & \\ VM_n & \mathcal{P}_{n/1}(t) & \mathcal{P}_{n/2}(t) & & \mathcal{P}_{n/m}(t) \end{array}$$

The degree of freedom on the state of VM_i is denoted by a binary variable $\mathcal{S}_{VM_i}(t)$ with $\mathcal{S}_{VM_i}(t) \in \{0, 1\}$.

$$\begin{cases} \mathcal{S}_{VM_i}(t) = 1 \quad \text{if the virtual machine} \\ VM_i \text{ is turned on at the time } t \\ \mathcal{S}_{VM_i}(t) = 0 \quad \text{if the virtual machine} \\ VM_i \text{ is turned off at the time } t \end{cases} \tag{2}$$

In the same way, the degree of freedom on the state of server S_j is denoted by a binary variable $S_{S_j}(t)$ with $S_{S_j}(t) \in \{0, 1\}$.

$$\begin{cases} S_{S_j}(t) = 1 & \text{if the server is turned on at the time } t \\ S_{S_j}(t) = 0 & \text{otherwise} \end{cases} \tag{3}$$

Contrains formulation

The optimization principle aims to place all the *VM's* containing applications on a minimum number of physical servers. Then idle servers can be turned off. They could be turned on if a peak load occurs and more physical machine are needed. Therefore, the optimization principle reduces the power consumption while guaranteeing constraints on the available resources and the quality of service (QoS/ SLA). So constraints may be formulated: constraint on the virtualization and overhead of *VM's*, on the state of server (on/off) and on QoS.

Virtualization and VM consolidation

For the creation of a new Virtual Machines, custom resource allocations is based on three parameters: shares, reservations and limits [20]. By server resource it is understood the server hardware specification (CPU, memory, power, storage, and network resources).

As default settings, resource allocations for a *VM's* is unlimited and is not reserved. They share resources equally when required resources are overcommitted. Note that the virtual machine performance is the only impact of resource allocations. When server resources are overcommitted, virtual machines always continues to run but their performance are degraded.

In detail, resource reservation specifies the guaranteed reservation (or minimum resources) for CPU in MHz or memory in MB for a virtual machine. Let us note that a virtual machine may be powered-on, only if the guaranteed reservation (CPU and memory) for the virtual machine is available, it means that the resources are not already reserved by other running virtual machines. For VM's to be deployed on a server S_i guaranteed resource constraints must be satisfied. So, is obtained:

- $CPU_{VM_{i/j}}$ is the guaranteed reservation of CPU for VM_i
- $CPU_{S_{j/j}}$ is the physical CPU of server S_j
- $MEM_{VM_{i/j}}$ is the guaranteed reservation for the memory for VM_i
- $MEM_{S_{j/j}}$ is the physical memory of server S_j

The above parameters are integer variables.

$$\begin{cases} \sum_{i=0}^{n-1}(CPU_{VM_{i/j}} \times \mathcal{P}_{i/j}(t)) \leq CPU_{S_j} \\ \sum_{i=0}^{n-1}(MEM_{VM_{i/j}} \times \mathcal{P}_{i/j}(t)) \leq MEM_{S_j} \end{cases} \tag{4}$$

The transfer function expressing the performance of each server is presented by equation 5. It is based on hosted application service on server. When there is no request $Rqst_i$ of application A_i during an anticipative period, VM_i is turned off.

The request of an application can be calculated with the help of relation 5.

$$\begin{cases} Rqst_{i/j}(t) \neq 0 \rightarrow \mathcal{S}_{VM_i}(t) = 1 \\ Rqst_{i/j}(t) = 0 \rightarrow \mathcal{S}_{VM_i}(t) = 0 \end{cases} \tag{5}$$

where $Rqst_{i/min}$ is the minimal request of the application A_i on the server S_j and $Rqst_{i/j}$ is the maximal request of the application A_i on the server S_j.

$$\begin{cases} Rqst_i(t) \leq Rqst_i(t) \times \mathcal{S}_{VM_i}(t) \\ Rqst_{i/j}(t) \geq Rqst_{i/min} \times \mathcal{S}_{VM_i}(t) \end{cases} \tag{6}$$

Let denote $U_{VM_{i/j}}(t)$ is the $\%CPU$ on the $VM_{i/j}$ which depends on the VM state $\mathcal{S}_{VM_i}(t)$. $U_{VM_{i/j}}(t)$ is a parameter which has an integer value. Is obtained:

$$\begin{cases} U_{VM_{i/j}}(t) = U_{i/j}(t) & \text{if} \quad \mathcal{S}_{VM_i}(t) = 1 \\ U_{VM_{i/j}}(t) = 0 & \text{if} \quad \mathcal{S}_{VM_i}(t) = 0 \\ U_{i/j}(t) \leq 100 \times N_{coresVM} & \\ U_{i/j}(t) \geq U_{VM_{i/j}idle} & \end{cases} \tag{7}$$

If a *VM* is active, it is placed only on a server at each time. This constraint is expressed by the sum of each row in hosting matrix.

$$\sum_{j=0}^{m} \mathcal{P}_{i/j}(t) \leq 1 \tag{8}$$

The number of *VM's* hosted on a server is given by:

$$N_{VMs/S_j} = \sum_{i=0}^{n-1} \mathcal{P}_{i/j}(t) \tag{9}$$

The hosting possibility depends on the one hand on the compatibility between the type of application and system architecture (compatible hypervisor, required hardware like type and number of CPUs, etc.) and on the other hand on the relation between the required resources of the *VM* and the available resources of the hosts at each anticipative period (available CPU, memory). For what concerning the architecture of the host, possible hosts for a *VM* are assigned by system administrators. If a server S_j is not able to hold a VM_i, it is assigned by administrator by adding constraint:

$$\mathcal{P}_{i/j}(t) = 0 \tag{10}$$

In this part, the virtualization overheads is considered. In term of energy consumption, overhead may be considered in starting a VM, shutting down a VM and moving a running *VM* to another server. Taking into account virtualization overheads allows reducing the number of migration which

TABLE I
VIRTUAL MACHINE STATE

	$\mathcal{S}_{VM_i}(t)$	$\mathcal{S}_{VM_i}(t+1)$	$SChg_{VM_i}(t,t+1)$
turn on	0	1	1
turn off	1	0	1
no change	1	1	0
no change	0	0	0

is usually limited by the migration flow of the network and the migration time. Moreover, it allows avoiding unnecessary migration in the situation where different solutions of *VM* deployment may lead to the same energetic gain, and mixed integer linear programming algorithm must choose a first found solution out of the possibilities. In this case is better to choose the solution with less migration. Since it is not easy to formulate a mixed integer linear programming problem distinguishing the number of starting, shutting down and migrating a *VM*, the proposed solution is to count the number of *VM* how are state changing between two consecutive anticipative periods (the value of $\mathcal{S}_{VM_i}(t)$ change from 1 to 0 or from 0 to 1 between two consecutive period) with the following hypotheses:

- For a long time horizon with many anticipative periods, the number of VM starting is equal to the number of VM shutting down.
- Moving a running *VM* between two different servers is considered as shutting down a VM on a server and starting simultaneous another VM on another server. So the migration consumption is equal to the sum of the starting consumption and shutting down consumption of a *VM*. A VM migration is equal to two *VM* state changes.

Let $SChg_{VM_i}(t,t+1)$ be the *VM* state changing between two times periodes t and $t+1$ which is expressed on the table I.

$$\begin{cases} SChg_{VM_i}(t,t+1)=1 & \text{if state (on/off) of } VM_i \\ & \text{is changed from } t \text{ to } t{+}1 \\ SChg_{VM_i}(t,t+1)=0 & \text{otherwise} \end{cases} \tag{11}$$

$SChg_{VM_i}(t,t+1)$ is computed by :

$$\begin{aligned} SChg_{VM_i}(t,t+1) = \mathcal{S}_{VM_i}(t) + \mathcal{S}_{VM_i}(t+1) \\ -2\times\mathcal{S}_{VM_i}(t)\times\mathcal{S}_{VM_i}(t+1) \end{aligned} \tag{12}$$

Using $\delta_{VM_i}(t) = \mathcal{S}_{VM_i}(t)\times\mathcal{S}_{VM_i}(t+1)$ with $\delta_{VM_i}(t)\in\{0,1\}$ eq. 12 will be transformed into a mixed integer linear programming from.

The *VM* state change overhead $P_{overheadVM_i}$ is equal to the average value of starting consumption P_{OnVM_i} and shutting down consumption P_{OffVM_i}. Finally, the global virtualization overhead $P_{OnOffVM}$ during the time horizon T (number of anticipative periods) is computed:

$$\begin{aligned} P_{OnOffVM} = \frac{(P_{OnVM_i}+P_{OffVM_i})}{2}\times \\ (\sum_t^{T-1}\sum_j^m\sum_i^n SChg_{VM_{i/j}}(t,t+1)) \end{aligned} \tag{13}$$

Turn On/Off physical servers
In addition to *VM* deployment strategy, other way to save power consumption is to turn on/off physical server. The first condition to turn of a physical server is that, the server doesn't have to host any *VM*. The optimization drives the server state and decides to turn off an idle server by comparing between the energy consumption of turning off and turning on it again with the energy consumption of server during the time where there is no hosted *VM* on server. First, it has to be determined the consumption to turn off and turn on again a physical server. Like virtualization task presented previously, it not easy to distinguish two actions of turning off and turning on a server. This total consumption is approximated via the product of the total of server state change $SChg_{S_j}(t,t+1)$ and the average value of starting consumption of server P_{OnS_j} and shutting down consumption of server P_{OffS_j} with the hypothesis that for a long time horizon with many anticipative periods, the number of action of starting a server S_j is equal to the number of action of shutting down a server S_j. In the same manner, the server state change $SChg_{S_j}(t,t+1)$ between two times periods t and $t+1$ is given by:

$$SChg_{S_j}(t,t{+}1) = \mathcal{S}_{S_j}(t){+}\mathcal{S}_{S_j}(t{+}1){-}2{\times}\mathcal{S}_{S_j}(t){\times}\mathcal{S}_{S_j}(t{+}1) \tag{14}$$

Using $\delta(t) = \mathcal{S}_{S_j}(t)\times\mathcal{S}_{S_j}(t+1)$ with $\delta_{S_j}(t)\in\{0,1\}$ the above equation will be transformed into a linear from.

The total energy consumption of turning On/Off servers during the time horizon t (number of anticipative periods) is given:

$$\begin{aligned} P_{On/OffServers} = \frac{(P_{OnS_j}+P_{OffS_j})}{2} \\ \times(\sum_t^{T-1}\sum_j^m SChg_{S_j}(t,t+1)) \end{aligned} \tag{15}$$

Constraint on the state of server during the time horizon t is then formulated: servers states are driven such as during each time interval where server is turned of, consumption of server without workload is grater then the energy consumption of turning off and turning on it again.

$$\begin{aligned} P_{On/OffServers} < \sum_t^T\sum_j^m(S_{S_j}(t) \\ \times P_{idle}\times\frac{T_{period}-T_{OnS_j}-T_{OffS_j}}{T_{period}}) \end{aligned} \tag{16}$$

with P_{idle} is the consumption in idle mode (without workload)}; T_{period} is period duration in second; $T_{OnS_j}$is the starting time of server S_j in second; $T_{OffS_j}$the shutting down time of server S_j in second. Let denote $P'_{S_j}(t)$, the energy consumption of the server $S_j$which depends on the server state $\mathcal{S}_{S_j}(t)$. So, is obtained:

$$\begin{cases} P'_{S_j}(t) = P_{S_j}(t) \text{ if } \mathcal{S}_{S_j}(t)=1 \\ P'_{S_j}(t) = 0 \text{ if } \mathcal{S}_{S_j}(t)=0 \\ P_{S_j}(t) \le P_{S_jmax} \\ P_{S_j}(t) \ge P_{S_jmin} \end{cases} \tag{17}$$

With P_{S_jmax} is the maximum power consumption of server, and $P_{S_jmin} = P_{standBy}+P_{virtualization}$, the total consumption of server in active state:

$$P_{ActiveServers} = \sum_{t}^{T} \sum_{j}^{m} (S_{S_j}(t) \times P'_{S_j}(t)) \tag{18}$$

QoS, response time constraint
The Service Level Agreement (SLA) is a part of a service contract that defines the required quality of a service between a data center administrator and a customer. These agreements are based on resources availability level, response time, server load, assigned resources, output bandwidth, etc. SLAs is also a way to indicate a customer priority level (such as Gold, Silver, or Bronze) in relation to other customer. These priorities can be used when resource allocation conflicts occur. SLAs provided by administrator depend not only on administrator service (maintenance) but also on the data center infrastructure which may be distinguished in different TIER ([2]). It includes IT system, cooling systems, power system, etc. In this optimization approach, the focus is on the virtualization mechanic within a pool of server while guaranteeing many constraints on the server resource allocation and respond time of server.

Constraint on priority level (or criticality level) of a hosted application is defined by hosting possibility constraint and the share and reservation resources expressed by constraints.

Processing time on a server (does not include network transfer time and wait time spent in the incoming throttling queues) depends on the server hardware (type of CPU, number of core, memory, frequency, etc), software, system environment (virtualization, non-virtualization) and the allocated resources to *VM* which hold the application.

Using the characterization of an application done in this section the constraints on the response time according to CPU occupation may be approximated as a linear function. Let $Rqst_{i/j}(t)$ is the workload (in requests/second) of application $A_i (A_i$is placed on $VM_i)$ at the time t, VM_i is hosted on server S_j, $\%CPU_j(t)$ is the total CPU occupation of the server S_j at the time t, is obtained:

$$\Longleftrightarrow \begin{cases} T_{rspA_i}(t) = \beta_{ij} \times \%CPU_j(t) \\ \beta_{ij} = \frac{T_{rspA_i max}}{90 \times N_{S_j}} \\ \%CPU_j(t) = \sum_{j=0}^{n} \%CPU_{i/j}(t) \end{cases} \tag{19}$$

$T_{rspA_i max}$ is the response times at saturation point (90% CPU) and $N_{S_j}$is the number of cores of server S_j. β_{ij} is defined for each combination $< A_i, S_j >$ because it depends both on the type of application A_i and on the architecture of S_j. Constraint on the response time for each application A_i aims to find the combination $< VM_i, S_j >$ such that the response time $T_{rspA_i}(t)$ for each request of A_i is lower or equal to the expected response time$T_{exptA_i}(t)$ at anticipative period t.

The objective function:
The final objective of this problem is to minimize the global consumption while respecting many constraint of QoS on the response time. Global energy consumption is computed by the sum of active server consumption given by eq. 15, virtualization consumption given by eq. 13 and the consumption of turning off/on servers eq. 18:

$$P_{global} = P_{On/OffServers} + P_{OnOffVM} + P_{ActiveServers} \tag{20}$$

IV. THE VALIDATION OF THE OPTIMIZATION PROBLEM

The problem can be divided into independent sub-problems. Each sub-problem related to a specific time horizon can be solved using mixed integer linear programming.

The validation scenario is composed of the optimization followed by the simulation step. Within the first step, the optimization, the problem (the cost function and the all the constraints stated earlier) is introduced in Matlab. The result is an optimal solution reflecting an allocation matrix of the virtual machines.

To validate the algorithm 5 types of servers and 5 types of application were used. In order to evaluate the proposed algorithm a commercial algorithm was replicated. Tables II and III presents the energy consumed by the 2 methods.

TABLE II
COMMERCIAL ALGORITHM - ENERGY CONSUMPTION

Hour	1	2	3	4	5
Server 1	660	632	445	335	529
Server 2	160	166	158	146	135
Server 3	142	180	180	178	173
Server 4	10	174	269	8	0
Server 5	5	244	250	272	7
Server Consumption	977	1396	1302	939	844
Total IT Consumption of the Data Center: 5458 kWh/hour					

TABLE III
THE PROPOSED ALGORITHM - ENERGY CONSUMPTION

Hour	1	2	3	4	5
Server 1	628	740	645	0	0
Server 2	0	0	0	140	0
Server 3	0	0	0	0	0
Server 4	0	250	261	250	25
Server 5	258	270	261	270	265
Server Consumption	886	1260	1167	660	290
Total IT Consumption of the Data Center: 4263 kWh/hour					

Table II shows that the commercial algorithm has a 5458 kWh/hour energy consumption versus a 4263 kWh/hour energy consumed by the depicted method. It can be stated that the proposed algorithm has a gain of approximately 22%.

V. CONCLUSIONS

Data centers are energy consumption system and a great need to increase their energy efficiency has been proven. This paper presents an algorithm that can be used to reduce energy consumption into a data center. The optimization is based on the behaviors of elements included within data center (different types of servers with different levels of availability and power consumption, as well as the global overview of the data center). These behaviors are modeled by transfer functions. The optimization scenario uses transfer functions to estimate the energetic effects of optimization activities and the predicted workload. In this paper, the considered problem, are formulated as a linear programming problem where all transfer/constraint functions and objective function are expressed under a linear form. Formulating the problem as a linear programming problem reduces the computation times which is usually limited by the anticipative time period. The final objective of this problem is to minimize the IT consumption while respecting many constraint of QoS on the response time. Global IT energy consumption is computed by the sum of active server consumption, virtualization consumption and the consumption of turning off/on servers.

REFERENCES

[1] Fan Xiaobo, Weber Wolf-Dietrich, and Barroso Luiz Andre. Power provisioning for a warehouse-sized computer. In *Proceedings of the 34th annual international symposium on Computer architecture*, ISCA '07, pages 13–23, New York, NY, USA, 2007. ACM.

[2] W. Pitt Turner IV, J. H. Seader, and K. G. Brill. Tier classifications define site infrastructure performance. *Uptime Institute White Paper*, 2010.

[3] Michael K. Patterson and Dave Fenwick. The state of data center cooling - a review of current air and liquid cooling solutions. Technical report, Intel Corporation White Paper, 2008.

[4] Das Rajarshi, Jeffrey O. Kephart, Lenchner Jonathan, and Hamann Hendrik. Utility-function-driven energy-efficient cooling in data centers. In *Proceeding of the 7th international conference on Autonomic computing*, ICAC '10, pages 61–70, New York, NY, USA, 2010. ACM.

[5] Victor K. Lee. Power and cooling architecture in future data center. *Emerson Network Power Journal*, 2010.

[6] Google. Our energy-saving data centers. Technical report, Google, 2010.

[7] Charles Lefurgy, Karthick Rajamani, Freeman Rawson, Wes Felter, Michael Kistler, and Tom W. Keller. Energy management for commercial servers. *Computer*, 12:39–48, 2003.

[8] Daniel Gmach, Jerry Rolia, Ludmila Cherkasova, and Alfons Kemper. Workload analysis and demand prediction of enterprise data center applications. *Proceedings of the Symposium on Workload Characterization (IISWC)*, 2007.

[9] Akshat Verma, Gargi Dasgupta, Tapan Kumar Nayak, Pradipta De, and Ravi Kothari. Server workload analysis for power minimization using consolidation. *Proceedings of the 2009 USENIX Annual Technical Conference*, 2009.

[10] Tibor Horvath, Tarek Abdelzaher, Kevin Skadron, and Xue Liu. Dynamic voltage scaling in multitier web servers with end-to-end delay control. *Computer*, 56 (4):444–458, 2007.

[11] Iulia Dumitru, Stéphane Ploix, Ioana Făgărăşan, and Sergiu Stelian Iliescu. Energy efficiency dependency analysis for a data center. In Ioan Dumitrache, editor, *Advances in Intelligent Control Systems and Computer Science*, volume 187 of *Advances in Intelligent Systems and Computing*, pages 307–321. Springer Berlin Heidelberg, 2013.

[12] Berral Josep Ll., Goiri Ínigo, Nou Ramón, Julià Ferran, Guitart Jordi, Gavaldà Ricard, and Torres Jordi. Towards energy-aware scheduling in data centers using machine learning. In *Proceedings of the 1st International Conference on Energy-Efficient Computing and Networking*, e-Energy '10, pages 215–224, New York, NY, USA, 2010. ACM.

[13] Luiz André Barroso and Urs Holzle. The case for energy-proportional computing. *Computer*, 40(12):33–37, 2007.

[14] Akshat Verma, Puneet Ahuja, and Anindya Neogi. pmapper : Power and migration cost aware application placement in virtualized systems. *International Conference on Middleware*, 2008.

[15] Themistoklis Charalambous and Evangelia Kalyvianaki. A min-max framework for cpu resource provisioning in virtualized servers using h1 filters. *Proceedings of the 49th IEEE Conference on Decision and Control*, 2010.

[16] Dara Kusic. Power and performance management of virtualized computing environments via lookahead control. *Cluster Computing*, 2009.

[17] Qian Zhu, Jiedan Zhu, and Gagan Agrawal. Power-aware consolidation of scientic workflows in virtualized environments. *Conference for High Performance Computing, Networking, Storage and Analysis*, 2010.

[18] Petrucci, Vinicius, Orlando Loques, and Daniel Mossé. A dynamic optimization model for power and performance management of virtualized clusters. In *Proceedings of the 1st International Conference on Energy-Efficient Computing and Networking*, e-Energy '10, pages 225–233, New York, NY, USA, 2010. ACM.

[19] G. Warkozek, E. Drayer, V. Debusschere, and S. Bacha. A new approach to model energy consumption of servers in data centers. *IEEE International Conference on Industrial Technology ICIT*, 2012.

[20] Inc. VMwareVMware. vsphere resource managemt. Technical report, VMware, 2011.

[21] SPEC. Specpower benchmark ssj2008. *http://www.spec.org/powersj200*, accessed on january 2011.

[22] Brian Renwick. Surprising cost savings through server room temperature monitor strategies. *The American Society of Heating, Refrigerating and Air-Conditioning Engineers (ASHRAE)*, 2009.