Received November 2, 2016, accepted November 21, 2016, date of publication December 1, 2016, date of current version January 4, 2017.

Digital Object Identifier 10.1109/ACCESS.2016.2633558

Energy Efficient Resource Allocation in Cloud Computing Environments

SHAHIN VAKILINIA[1], (Student Member, IEEE), BEHDAD HEIDARPOUR[2], AND MOHAMED CHERIET[1], (Senior Member, IEEE)
[1]Department of Computer Engineering, École de technologie supérieure, Montreal, QC H3C 1K3, Canada
[2]École de technologie supérieure, Montreal, QC H3C 1K3, Canada
Corresponding author: S. Vakilinia (shahin.vakilinia@synchromedia.ca)

ABSTRACT Power consumption is one of the major concerns for the cloud providers. The issue of disorganized power consumption can be categorized into two main groups: one caused by server operations and one occurred during the network communications. In this paper, a platform for virtual machine (VM) placement/migration is proposed to minimize the total power consumption of cloud data centers (DCs). The main idea behind this paper is that with the collaboration of optimization scheduling and estimation techniques, the power consumption of DC can be optimally lessened. In the platform, an estimation module has been embedded to predict the future loads of the system, and then, two schedulers are considered to schedule the expected and unpredicted loads, respectively. The proposed scheduler applies the column generation technique to handle the integer linear/quadratic programming optimization problem. Also, the cut-and-solve-based algorithm and the call back method are proposed to reduce the complexity and computation time. Finally, numerical and experimental results are presented to validate our findings. Adaptation and scalability of the proposed platform result in a notable performance in VM placement and migration processes. We believe that our work advances the state of the art in workload estimation and dynamic power management of cloud DCs, and the results will be helpful to cloud service providers in achieving energy saving.

INDEX TERMS Cloud computing, optimization, integer linear/quadratic programming, column generation, dynamic resource allocation, estimation theory, time-varying Kalman filter.

I. INTRODUCTION

Cloud computing has already revolutionized traditional Information Technology industry through helping developers and companies overcome the lack of hardware capacity (e.g. CPU, Memory, and Storage) by allowing the user to access on-demand resources through the Internet. The widespread employment of cloud Data Centers (DCs) necessitates the cloud providers(e.g., Amazon Rackspace) to improve cloud efficiency regarding the operational costs. Energy consumption is the key concern in operational costs of cloud systems. With the growing number of in-service servers, the global expenditure on enterprise energy usage and server cooling is estimated to be considerably high [1]. Based on recent research outcomes, up to 20% savings can be achieved on the energy consumptions of DCs. These savings lead to an additional 30% saving on cooling energy requirements [2].

Dynamic power management techniques aim to reduce the energy wastage in DCs by temporarily shutting servers down when they are not required. It also applies power saving technologies, such as Dynamic Voltage and Frequency Scaling (DVFS), to minimize the power level of active servers [3]. However, setup and transition times delay of the full reactivation or switching the power level of a server can adversely affect the system performance. Hence, to be able to dynamically manage the number of active servers and their performance level, the amount of incoming workload and their requirements should be estimated precisely. The total workload of DC consists of several jobs, and each job includes several Virtual Machines (VMs). The VMs of incoming jobs should be assigned to the active servers. Concurrently, one should take into account the role of all server resources namely CPU, memory, and storage in VM placement process. As a result, this will become a multidimensional bin packing problem.

Based on the types of applications served by the cloud computing center, there is a vast diversity in the demand resource profiles. In general, computing tasks such as web serving are more process intensive, while database operations typically require high-memory support. One of the other essential characteristics of a cloud computing system is diversification of server resources as well as the types of workloads. As time goes by, DCs update the configuration of their resources, the processing capabilities, memory and storage spaces. They also construct new platforms based on the new high-performance servers while the older servers are still operational. Due to heterogeneity of both servers and workloads, designing an optimal resource allocation algorithm concerning energy and cost efficiency becomes very complicated.

Beside power usage of servers, communication also impacts both performance and power consumption of the operations. Communication increases the job execution latency and the power consumption. One way to mitigate the Cloud Network (CN) power usage is to apply traffic aware VM placement methods [7]–[9]. Nevertheless, due to high variety, dynamicity, and heterogeneity of workload characteristics, traffic awareness is almost impossible on practical solutions and therefore DC traffic approximation should be applied.

All in all, the formulation of VM placement problem would include both network and servers power usages. In this paper, a platform for VM placement and migration in the DC that minimizes the power consumption of the DC is proposed. First, the incoming workload regarding the number of different types of jobs and different number of VMs are predicted for the next time slot. Secondly, the problem of VM placement and migration for power minimization, which is NP-hard [10], will be solved according to the estimation and available resources. Next, column generation (CG) technique is used to solve this large-scale optimization problem. Moreover, depending on the time limit and complexity constraints, three methods of off-line pattern generation, cut and solve [13], [14] , and Call-Back [32] are also proposed for initiation, limiting the searching area, and optimization termination, respectively. These methods are added to mitigate the complexity order of the optimization problem further. The main contributions of this paper are as follows:

- Heterogeneous resources and workloads of a DC are modeled and power efficient network-aware resource allocation platform is proposed to optimize the power consumption of cloud data centers.
- Auto Regressive Integrated Moving Average (ARIMA) based Kalman Filter (KF) is proposed to estimate the incoming workload and prediction error is also considered in the optimal resource allocation.
- CG technique is utilized in dynamic job scheduler with optimization of cloud power consumption. Then, offline pattern initiation, cut and solve method and call back approach are proposed to reduce the complexity and search space and to make it scalable regarding the scheduling deadline.

The remainder of this paper is organized as follows: Related work is discussed in section II. Section III introduces the notations and preliminaries of the cloud computing DC model. Section IV describes the job types of cloud DC. In section V, suggested platform is propounded. Also, the details of the estimation process and scheduling, which includes the discussion of the optimization in scheduling modules. Section VI introduces CG and discusses initialization, cut and solve, and heuristic algorithm for immediate termination. In Section VII, we give a comparison of numerical and experimental results with the closest related works that have been referred to in Section II. Finally, Section VIII concludes the paper and introduces the possible future work.

II. RELATED WORK

Despite the ubiquitous research attention devoted to power efficient resource allocation in cloud computing systems, it lacks from the optimal dynamic power management practical platforms. Dynamic power management technique necessitates a forecast of the workload of cloud computing DC. Some research papers such as [23], have studied stochastic modeling of cloud computing systems to predict the available resources and the workload of the DC. However, either exact analysis retain the restrictive distributions such as Poisson and Exponential, are used for the arrival and departure rates of the cloud workload or the accuracy of the analysis is degraded by some approximations.

Different predictive policies attempt to predict the request rate and to track the future loads of the DC. Conventional dynamic power management approaches, e.g., [14]–[16] use prediction policies such as Moving Average (MA) and Linear Regression (LR). In MA method, the request rates are averaged over a time window to predict the future job arrival rate. LR method is identical to MA except for the estimation of the request rate, which is made by matching the best linear fit to the values in the window. The best forecasting result with the highest accuracy is achieved using the Auto Regressive Integrated Moving Average (ARIMA) technique [18], [19], and [22]. In time series analysis of non-stationary scenarios, it is preferable to use an ARIMA model, which is a generalization of MA model fitted to the time series data. Calheiros *et al.* [22] applied prediction module based on ARIMA model to estimate the requests for the application servers of SaaS providers and later evaluated the accuracy of the future workload prediction using real traces of requests to web servers from the Wikimedia Foundation. The average accuracy of ARIMA is measured up to 91 percent. Assuming that the number of running tasks is a stationary process, Zhang *et al.* [18] also used ARIMA model-based estimator to predict the arrival rate and the number of long running tasks when the trend of resource demand is stable. Zhang *et al.* continued their analysis in [18], by using real traces obtained from Google compute clusters, indicate that the prediction Root Square Error (RSE) of ARIMA in the large scale is less than one percent.

Zhang *et al.* [17], also addressed the heterogeneity of workloads and PMs. According to their characteristics, tasks are classified into classes with similar resource demands and performance characteristics. Different types of servers are also considered based on their platform ID and capacities on various resources. An estimator based on time series has been implemented to predict workload rate. Then, a heterogeneity-aware resource monitoring and management system dubbed Harmony was proposed to perform dynamic capacity provisioning to minimize the total energy consumption and scheduling delay considering heterogeneity as well as reconfiguration costs.

In this paper, taking the same approach as in [17] and [21], an estimator is used to estimate the arrival rate of the new jobs in the system. However, non-stationary space of the job arrival process results in such a high level of error in which the model is rendered unreliable for application in heterogeneous scenarios. Therefore, to optimally manage the resources, it is better to consider the prediction error of the load more precisely. In this paper, state-space of KF is used to predict the workloads of the DC in the presence of non-linear structural changes and irregular patterns. A time series ARIMA is employed to obtain the best initial parameters of the Kalman model [33]. So, KF is applied on ARIMA model to reduce the prediction error of arrival rate. KF is popular due to its desirable non-linear performance. Incorporating non-linear effects of variables, structural breaks can be easily identified with state space than simple ARIMA.

Moreover, the estimation error is also considered in the resource allocation problem by reserving some resources for the unpredicted load as mentioned in [6]. To the best of our knowledge, dynamic resource management in [6] is the only technique to scale the DC with the unpredictably changing load. It should also be noted that performance of ARIMA was evaluated for Google Compute real cluster trace [17] and Wikimedia webservers request [21] and estimation error rate of ARIMA enhances for the general large scale scenarios while the ARIMA-based KF estimator proposed in this paper targets heterogeneous type of workloads.

As opposed to the workload request estimation, the forecast of the exact traffic among the VMs allocated to the DC is very complicated in practice, if not impossible, due to the high variety of the cloud network traffic. Therefore, the traffic rate should be approximated. Li *et al.* [10] and You *et al.* [19] associated the network cost with the number of separated VMs of tenants by defining different cost functions in which the number of job fragmentations is the variable. Reference [10] and [19] used a single-dimensional resource allocation algorithm and set a slot to represent one resource unit (CPU/memory/disk) in a way that each slot can host one VM. You *et al.* [19] also proposed a binary search-based heuristic algorithm to achieve an optimum point in the trade-off between PM-cost and network cost to minimize the cost according to the arbitrary assumption for the proposed cost functions. Reference [19] proposed an optimal solution to reduce the network cost for a homogeneous scenario by demonstrating that the most active VMs has to be placed on the PMs with the higher capacity. Similarly, in this paper, the network power consumption is attributed to the number of separated VMs of a tenant on each server. According to the results of [19], the proposed cut and solve method prioritizes the PMs with the higher capacity in the search area. However, instead of an unrealistic homogeneity assumption, as in [10] and [19], in this paper, heterogeneity of both workload and machine hardware are considered in the scheduling problems.

Assi *et al.* [20] addressed the issue of traffic in data center networks from a different aspect. Assi *et al.* [21] assumed that each job is characterized by a set of VMs communicating with each other. The problem of mapping traffic flows of each job into VLANs and selecting the most efficient spanning tree protocols with the objective of load balancing is investigated regarding the bandwidth requirements of VMs and bandwidth constraints. CG technique is proposed to solve the optimization problem reducing the complexity and search space and then a semi-heuristic decomposition approach is proposed to make it scalable. In this paper, similar to [20] CG approach is taken into account to solve the optimization problem. However, while solving the optimization problem of typical cloud VM placement [22] took more than few hours, the time needed to reach the solution can decrease to few minutes when the cut and solve technique and the Call-Back method are applied. Moreover, it is worth mentioning that the proposed platform is independent of the DC topology.

The work in this paper addresses various challenges of the research mentioned above in such areas as heterogeneity, the power consumption of DC, and workload estimation to present a robust method that can generate more overall and reliable outcomes.

III. PRELIMINARIES AND NOTATIONS

We assume that a DC has T types of servers, where each server type is determined by the amount of various kinds of resources that it contains. Note that assumption of T servers address the heterogeneity of the resources at DC. A server type may have K different types of resources such as bandwidth, storage, CPU, and memory. A unique resource vector determines the amount of resources that each server type has. Let M_t denote the number of type t servers in the DC where $t \in \{1, , T\}$. It is also assumed that c_t^k denote the capacity of type t servers on type k resource.

The power consumption of an *on* type t server will be denoted by Q_t. R different VM configurations are assumed. Each VM configuration is determined by the amount of various types of resources it contains. Let i_r^k denote the type k resource requirement of the type r VM. According to the job requirements, it is also assumed that there are H different types of jobs, where each job type requires a random number of VMs from different types. Assuming H various types of jobs addresses heterogeneity of the incoming workloads to DC.

Due to the dynamicity and time variation, data related to the previous W slots are measured and stored in the platform. So, W is the window size, and the most recent data belong to W slot before are captured. The historical data from

$w \in \{1, \ldots, W\}$ slot before will be used to estimate the number of jobs and the attributed number of VMs. In other words, W represents both degree of differencing and the order of the Moving-average of the ARIMA model.

We let $N_{h,\ell-w}$, $V^r_{h,\ell-w}$ represent the total number of type h jobs and the total number of type r VMs dedicated to type h jobs at the $\ell - w$ time slot (lag w), respectively.

To optimally allocate resources among the jobs, $N_{h,\ell}$ and $V^r_{h,\ell}$ should be estimated using data from previous slots. To simplify the notations, $N_{h,\ell}$ and $V^r_{h,\ell}$ are summarized by N_h, V^r_h in Section V.

Let also P_h denote the communication power consumption between two VMs of the type h job. The scheduling variable $x^{m_t}_{r,n_h}$ represents the number of type r VMs in m^{th} type t server assigned to serve job n_h where $m_t = \{1_t, \ldots, M_t\}$. It is desired to find the optimal values of $x^{m_t}_{r,n_h}$s that minimize the DC power consumption. Similarly, connectivity variable xÌČ$^{m_t}_{n_h}$s is defined as the number of VMs assigned to job n_h on the m^{th} type t server.

The notation for the mathematical model has been summarized in Table 1.

IV. MODELING OF THE CLOUD COMPUTING JOBS

The current model assumes a varying number of jobs in a cloud computing DC in different time slots. Each job may require different number of VMs, which may be assigned to several servers. To minimize communication among VMs, it is preferable to place all the VMs on the same PM or PMs close to each other. However, to reduce the number of servers, VMs of jobs may distribute among several servers. Thus, there is always a trade-off between network communication and the power consumption of the servers[10]. In this research, two different types of jobs, namely centralized and distributed, are investigated.

Let us bring an example for further explanation about this trade off. Assume that there are three active servers in the DC and other servers are shut down, and hypervisors of these servers can accept only one more VM. In this situation, a job demanding 3 VMs arrives at the DC. Under this circumstances, there are two main resource allocation strategies. It is better to distribute a VM of the incoming job to each server to minimize the server power consumption while to reduce the power consumption associated with network communication, it is better to turn on a new server and allocate all the VMs of the incoming job to the new server. Generally speaking, power consumption is related to both network and servers. Thus, the optimal solution varies over the time.

In the first scenario, corresponded to [24]–[26], each centralized job has a centralized database and VMs of the job have to communicate with the main database to serve their tasks. As a result, the VMs assigned to a job on different servers need to communicate with a database. Then, a distributed model is investigated so that all the VMs of a job communicate with each other to serve their tasks [28], [29]. Similarly, the number of fragmentations (i.e., the number of servers containing VMs of a job) is linearly correlated with the communication rate of the job. Hence, the resource allocation problem can be modeled linearly. However, in the distributed model, communication rate of a job is approximated in a quadratic format [22].

TABLE 1. Table of parameters.

Parameters	Indicator
R	number of VM types
K	number of resource types
T	number of different types of servers
H	number of different types of jobs
i^r_k	type k resource requirement of type r VM
c^k_t	capacity of type t servers over type k resource.
Q_t	power consumption of type t server
M_t	number of type t servers in the datacenter
$a_{h,\ell}$	parameter of autoregressive terms of ARIMA model related to ℓ^{th} time slot and job type h
$b_{h,\ell}$	parameters of moving average terms of ARIMA model related to ℓ^{th} time slot and job type h
W	the window size, order of the Moving Average in ARIMA
P_{n_h}	communication power consumption between VMs of a type h job
θ	Integer number much larger than the maximum value of scheduling variables integer (here is equal to 10000)
$\xi_{h,\ell}$	approximation error of KF
$\Lambda_{h,\ell}$	covariance error of approximation error of KF $\xi_{h,\ell}$
$N_{h,\ell}$	number of type h jobs in the datacenter at ℓ^{th} time slot
$N_h, \hat{N}_{h,\ell}$	Estimated number of type h jobs in the datacenter at ℓ^{th} time slot
$\eta_{h,\ell}$	the prediction error $\hat{N}^\lambda_{h,\ell} - N^\lambda_{h,\ell}$
$\psi_{h,\ell}$	the covariance matrix of the prediction error ($\eta_{h,\ell}$)
$V^r_{h,\ell}$	number of assigned VMs to type h jobs at l^{th} time slot
$v^r_{n_{h,\ell}}$	number of type r VMs required by job n_h at ℓ^{th} time slot
$x^{m_t}_{r,n_h}$	number of type r VMs in m_{th} type t server assigned to serve job n_h
$\widetilde{x}^{m_t}_{n_h}$	number of VMs assigned to job n_h on the m^{th} type t server
$z^{m_t}_{n_h}$	binary variable representing the connectivity of n_h job to m_{th} type t server
f_{n_h}	number of pieces of job n_h
y_{m_t}	binary variable *ON* on or *OFF* status of m_{th} type t server
$\beta^{m_t}_{r,n_h}$	binary variable representing whether r VMs in m_{th} type t server assigned to serve job n_h migrated or not.
$e^{m_t}_{r,n_h}$	the number of type r VMs of unpredicted load of job n_h allocated on server m_t.
$\widetilde{e}^{m_t}_{n_h}$	total number of VMs required by the unpredicted load of job n_h allocated on server m_t.
$\Lambda_{h,\ell}$	the covariance matrix of the KF approximation error
G_r	the power consumption related to the migration of type r VMs
J_t	Number of possible patterns for type t server.
$x^{j_t}_{r,n_h}$	number of VM type r of the job n_h over the server type t by pattern j_t.
$\widetilde{x}^{j_t}_{n_h}$	Total number of VMs of the job n_h over the server type t by pattern j_t.
m_{j_t}	number of server type t is required with the configuration j_t in order to serve the jobs
$w^{j_t}_{n_h}$	binary variable representing the connectivity of n_h job to type t server with pattern j_t

Fig. 1 represents the placement and connection of VMs demanded by these two types of jobs in the cloud DC. As it is shown, incoming jobs are heterogeneous regarding demanding the different number of VMs from various kinds.

In this figure, it is assumed three types of VMs ($R = 3$) in three colors (gray, blue and green). There are two types of jobs ($H = 2$) and there is a job from each type in the

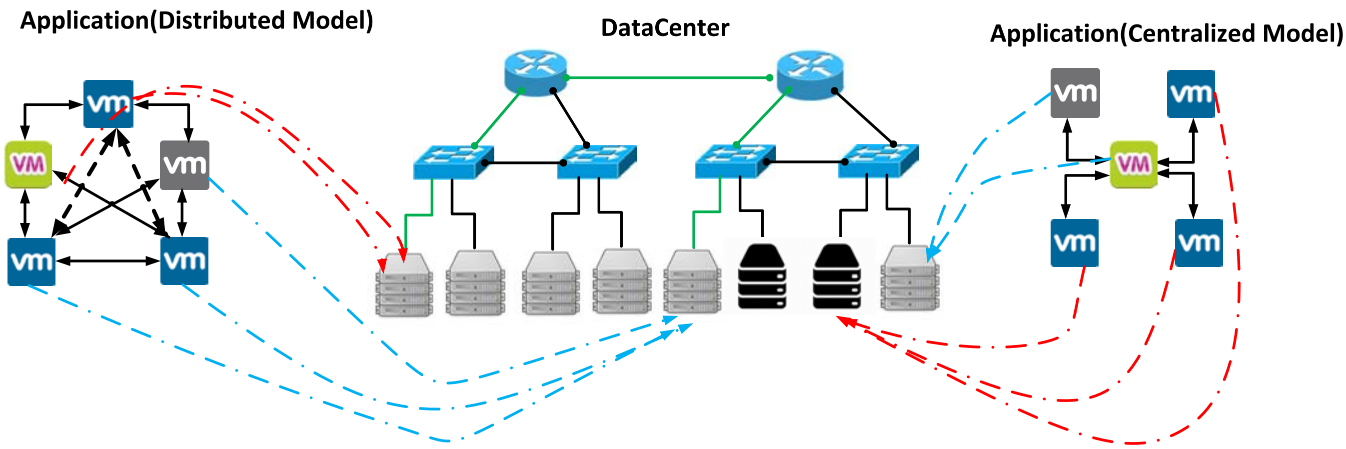


FIGURE 1. Example of VM Allocation in the DC.

system ($N_{h=1} = 1, N_{h=2} = 1$). It is also assumed that there are two types of servers ($T = 2$), in white and black colors. It is assumed that the resource allocation algorithm assigns 2 VMs of the distributed application to a white server and three others are assigned to another server located somewhere else at the DC. The network power consumption of the VMs assigned to one server is zero while according to the DC topology, there would be a power consumption associated with the network communication for the VMs located in different servers. Fig. 1 shows these communication link with the green line. For the centralized application, the scenario is different. If the VM assigned to the same server as database VM (green one) is allocated the communication power consumption is zero (For instance, there is no network power consumption between gray VM and green one) while if they are assigned to different servers (blue VMs), there would be a communication energy consumption.

V. SYSTEM PLATFORM OVERVIEW

In this section, the architecture for VM provisioning module is described as it is depicted in Fig. 2. As shown in this figure, the estimator, and the estimation error updater modules predict the load and prediction error. The predicted data is then delivered to the scheduler modules. Historical information of incoming workloads, i.e. the number of jobs and their associated VMs from W time slot before, in previous time slots are used to update and tune the workload estimators and the estimation error.

The estimator module predicts the incoming load in terms of some jobs requiring a different number of VMs for the next time slot. ARIMA-based KF is proposed to predict the total number of incoming VMs and jobs for the next time slot. However, it is convenient that incoming job arrival is a random process and only the expected values can be reached. Therefore, the prediction error is inevitable which has to be taken into account in the resource allocation procedure. The details of estimator module will be discussed in Subsection V-A.

Moreover, another module is needed to monitor the workload and resources in the DC and to gather the information about the availability of the resources inactive servers.

After predicting the load and monitoring the available resources, the incoming load should be scheduled. The proposed scheduling module consists of two schedulers for expected and unexpected loads. As it mentioned earlier unexpected load refers to the estimation error of prediction module. Schedulers of the expected and unexpected loads solve the power consumption minimization problem to distribute the load among the servers. First, using CG, the optimization problem is solved for the expected incoming load and previous loads in the system. Then, the scheduling variables are used as inputs for the other optimization problem to reserve some capacity for the unexpected loads. Finally, the variables related to the available resources for the next time slot are updated.

According to the result of the optimizations, the capacity provisioning module adds/drops resources by turning *on/off* the servers. It also assigns the workloads in terms of VMs to the activated servers and migrates some old VMs into the other servers.

A. ESTIMATION TECHNIQUE

In cloud computing, applications compete for resources. By causing the host load to vary over time, this competition renders the load prediction very complicated. The previous literature on the forecasting of the cloud workload and available resources includes time series prediction based on historical information captured throughout monitoring of the systems. In this paper, first, the workload prediction module based on ARIMA model has been developed to approximate the incoming workloads of different jobs regarding VMs. Setting ARIMA model coefficients may have some approximation error represented by vector $\xi_{h,\ell}$ with its attributed covariance error indicated by $\Lambda_{h,\ell}$.

Then, the estimation is obtained recursively by the well-known KF to decrease the forecasting error [29].

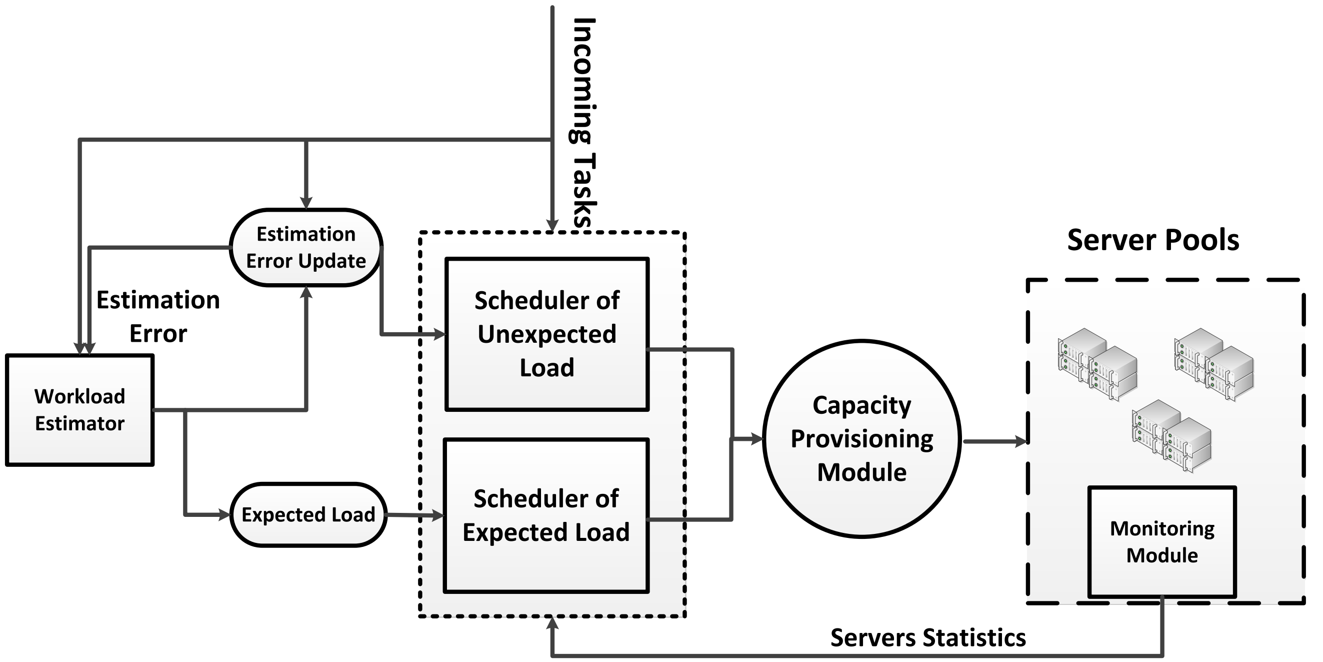


FIGURE 2. The Proposed Platform Framework.

The estimated number of incoming type h jobs at the next time slot, represented by $\hat{N}^{\lambda}_{h,\ell}$, is obtained by the following equation:

$$\hat{N}^{\lambda}_{h,\ell} + \sum_{w=1}^{W} a_{h,\ell-w} N^{\lambda}_{h,\ell-w} = \sum_{r=1}^{R} \sum_{w=1}^{W} b^{r}_{h,\ell-w} V^{r,\lambda}_{h,\ell-w} \quad (1)$$

In Eq. (1), $\hat{N}^{\lambda}_{h,\ell}$ is dependent of the number of previous jobs and the previous VMs types in the DC. $V^{r,\lambda}_{h,\ell-w}$ also represents the estimation of the total number of incoming type r VMs of type h jobs at time slot $\ell - w$. W is the window size (order of the moving-average) calculated by the auto-correlation function of the number of jobs and VMs over the time series [30]. It is also assumed that $\eta_{h,\ell}$ represents the prediction error $\hat{N}^{\lambda}_{h,\ell} - N^{\lambda}_{h,\ell}$ with $\psi_{h,\ell}$ as the covariance. Main target of KF is to predict $N^{\lambda}_{h,\ell}$s and $V^{r,\lambda}_{h,\ell}$s.

$$\hat{N}_{h,\ell} = \Theta_{h,\ell}\pi_{h,\ell} + \xi_{h\ell} \quad (2)$$
$$\pi_{h,\ell+1} = \pi_{h,\ell} + \eta_{h,\ell} \quad (3)$$

where,

$$\Theta_{h,\ell} = [N^{\lambda}_{h,\ell-1}, \ldots, N^{\lambda}_{h,\ell-W}, V^{1,\lambda}_{h,\ell-1}, \ldots, V^{r,\lambda}_{h,\ell-w}, \ldots, V^{R,\lambda}_{h,\ell-W}] \quad (4)$$
$$\pi_{h,\ell} = [a_{h,\ell-1}, \ldots, a_{h,\ell-W}, b^{1}_{h,\ell-1}, \ldots, b^{r}_{h,\ell-w}, \ldots, b^{R}_{h,\ell-W}] \quad (5)$$

$\pi_{h,\ell}$ denotes the KF state space vectors equal to ARIMA coefficients, which should be updated as follows:

$$\hat{\pi}_{h,\ell+1|l} = (I - L_{h,\ell})\hat{\pi}_{h,\ell|\ell-1} + L_{h,\ell}\Theta_{h,\ell}\pi_{h,\ell} \quad (6)$$

where,

$$L_{h,\ell} = \Sigma_{h,\ell|l-1}\theta_{h,\ell}(\psi_{h,\ell} + \theta_{h,\ell}\Sigma_{h,\ell|l-1}\theta^{T}_{h,\ell})^{-1} \quad (7)$$

where, $\Sigma_{h,\ell|l-1} = E[(\pi_{h,\ell} - \hat{\pi}_{h,\ell|l-1})(\pi_{h,\ell} - \hat{\pi}_{h,\ell|l-1})^{T}]$ and should be upgraded as follows:

$$\Sigma_{h,\ell+1|l} = \Sigma_{h,\ell|l-1} - \Sigma_{h,\ell|l-1}\theta_{h,\ell}(\psi_{h,\ell} + \theta_{h,\ell}\Sigma_{h,\ell|l-1}\theta^{T}_{h,\ell})^{-1} \times \theta^{T}_{h,\ell}\Sigma_{h,\ell|l-1} + \Lambda_{h,\ell} \quad (8)$$

As it mentioned earlier, $\Lambda_{h,\ell}$ and $\psi_{h,\ell}$ are the covariance matrix of the approximation error and prediction error of the type h workload at time slot l. For more details and please check [29]. The estimation of the number of various kinds of jobs and their associated number of VMs helps to activate the servers proactively so as to avoid the delay in setup time of the servers, which can adversely affect the system performance.

B. SCHEDULING OF THE EXPECTED LOAD

The total power consumption of the cloud DC is minimized when the job load is served by the minimum number of servers, and each job is assigned VMs from as few servers as possible. In this subsection, the optimization problem and CG both for centralized and distributed models are added. First, typical ILP/IQP are used to model and solve the optimization problem. Second, CG will be introduced to reduce the complexity of the optimization problem.

1) CENTRALIZED MODEL

From the definitions in Table 1., the following relationships exist for Centralized Model (CM): $\forall n_h \in \{1_h, \ldots, N_h\}$, $\forall h \in \{1, \ldots, H\}$, $\forall m_t \in \{1_t, \ldots, M_t\}$, $\forall t \in \{1, \ldots, T\}$,

$\forall r \in \{1, \ldots, R\}$

$$\widetilde{x}_{n_h}^{m_t} = \sum_{r=1}^{R} x_{r,n_h}^{m_t} \tag{9}$$

$$z_{n_h}^{m_t} = \begin{cases} 1 & if\ \widetilde{x}_{n_h}^{m_t} > 0 \\ 0 & \widetilde{x}_{n_h}^{m_t} = 0 \end{cases} \tag{10}$$

Eq. (10) extracts the connectivity variables, $z_{n_h}^{m_t}$, out of the scheduling variables, $x_{n_h}^{m_t}$s. Then, the communication power usage of a job n_h is approximated and associated with the total number of pieces of the job in the form of VMs.

$$f_{n_h} = \sum_{t=1}^{T} \sum_{m_t=1_t}^{M_t} z_{n_h}^{m_t} \tag{11}$$

f_{n_h} represents the number of pieces of job n_h. Let binary variable y_{m_t} denote *on* or *off* status of m^{th} type t server,

$$y_{m_t} = \begin{cases} 1 & if\ \sum_{h=1}^{H} \sum_{n_h=1_h}^{N_h} z_{n_h}^{m_t} > 0 \\ 0 & \sum_{h=1}^{H} \sum_{n_h=1_h}^{N_h} z_{n_h}^{m_t} = 0 \end{cases} \tag{12}$$

Eq. (12) helps to find server status variables, y_{m_t}s, using $z_{m_t}^{m_t}$ connectivity variables. Accordingly, the optimization problem is given by:

$$\min_{x_{r,n_h}^{m_t}} \sum_{h=1}^{H} P_h \sum_{n_h=1_h}^{N_h} f_{n_h} + \sum_{t=1}^{T} Q_t \sum_{m_t=1_t}^{M_t} y_{m_t}$$

S.T. : (9), (10), (11), (12)

$$\sum_{t=1}^{T} \sum_{m_t=1_t}^{M_t} x_{r,n_h}^{m_t} \geq v_{n_h}^{r}$$

$$\sum_{h=1}^{H} \sum_{n_h=1_h}^{N_h} x_{r,n_h}^{m_t} i_t^k \leq c_t^k \tag{13}$$

In the objective function, the first and second terms correspond to the communications and server power consumptions of the datacenter respectively. The first group of constraint ensures that VM requirements of each type of job are satisfied and the second group guarantees that resource demands of jobs scheduled on a server do not exceed the resource capacities of that server. In order to linearize the constraints (10) and (12) in previous page, they are substituted with the following constraints:

$$\begin{aligned} \widetilde{x}_{n_h}^{m_t} - z_{n_h}^{m_t} &\geq 0 \\ \theta z_{n_h}^{m_t} - \widetilde{x}_{n_h}^{m_t} &\geq 0 \\ \sum_{h=1}^{H} \sum_{n_h=1_h}^{N_h} z_{n_h}^{m_t} - y_{m_t} &\geq 0 \\ \theta y_{m_t} - \sum_{h=1}^{H} \sum_{n_h=1_h}^{N_h} z_{n_h}^{m_t} &\geq 0 \end{aligned} \tag{14}$$

θ denotes an integer much larger than the maximum value of the above positive integer. For the remainder of the paper, this replacement will be referred as positive Integer to Binary Linear Conversion (IBLC) constraints.

2) DISTRIBUTED MODEL

We assume a Distributed Model (DM), where a job may be assigned VMs on different servers. There will be a need for communications among the VMs assigned to a job on different servers. This demand is proportional to the product of the number of VMs assigned to each job on each pair of servers. Similarly, it is desired to find the optimal values of $x_{r,n_h}^{m_t}$s that minimize the DC power consumption. As a result, the optimization objective is given by;

$$\min_{x_{r,n_h}^{m_t}} \sum_{h=1}^{H} P_h \sum_{n_h=1_h}^{N_h} \sum_{t=1}^{T} \sum_{m_t=1_t}^{M_t} \{ \sum_{t'=1}^{T} \sum_{m'_{t'}=1_{t'}}^{M_{t'}} x_{r,n_h}^{m_t} x_{r,n_h}^{m'_{t'}} - (x_{r,n_h}^{m_t})^2 \}$$
$$+ \sum_{t=1}^{T} Q_t \sum_{m_t=1_t}^{M_t} y_{m_t}$$

In the above objective function, the first and second terms correspond to the communications and server power consumptions of the DC, respectively. As shown, the energy consumption attributed to the VMs communication is approximated as a linear function of the total number of communication links to the jobs.

C. DYNAMIC JOB SCHEDULING

In this section job scheduling is extended by considering the optimization of power consumption as a function of time. As a result, it is assumed that time-axis is slotted and VMs are assigned to jobs in time slot unit. Also, the job scheduling is considered in such a way to allow VM migration. In other words, the analysis is extended to the case where location of the VMs of different jobs varies over time.

Let us consider n_h^{th} job, which is in the system in the current slot and will continue to receive service in the next slot. Let $x_{r,n_h}^{m_t}, x_{r,n_h}^{\prime m_t}$ denote the number of type r VMs assigned to this job over the m^{th} type t server during the current and next slots respectively. The following binary variables are defined,

$$\beta_{r,n_h}^{m_t} = \begin{cases} 1 & if\ (x_{r,n_h}^{m_t} - x_{r,n_h}^{\prime m_t}) < 0 \\ 0 & otherwise = 0 \end{cases} \tag{15}$$

The value of $\beta_{r,n_h}^{m_t}$ shows whether the type r VMs required by job n_h have migrated or not. In the case of VM has migrated from the m_t server, $\beta_{r,n_h}^{m_t}$ will have a nonzero value and in all other cases a zero value. The objective function of this optimization problem is given by:

$$Min\{(13) + \sum_{h=1}^{H} \sum_{n_h=1_h}^{N_h} \sum_{r=1}^{R} G_r \sum_{t=1}^{T} \beta_{r,n_h}^{m_t} |x_{r,n_h}^{m_t} - x_{r,n_h}^{\prime m_t}|\} \tag{16}$$

where the absolute value of $(x_{r,n_h}^{m_t} - x_{r,n_h}^{\prime m_t})$ corresponds to the number of VM migrations. In the above, migration of a VM is allowed if it results in power saving larger than the power cost of the migration. Job scheduling without VM migration can be achieved by setting G_r, i.e., the power consumption related to the migration of type r VMs, to an enormous value. This will prevent migration as any power saving can not offset its cost. As a result, old jobs will preserve their VM assignments.

Moreover, in order to linearize Eq. (15) similar to Eq.(14), IDBLC is applied and the associated constraints are added into the problem.

D. COLUMN GENERATION

The scheduling problem in its current form is NP-hard. For large scale DCs, finding the global optimum point of an ILP becomes overly complicated and time-consuming. Due to the similarity of the current problem with the cutting-stock problem, a well-known CG technique is used to solve the problem. In this subsection, the application of CG technique as a method to reduce the search space of the optimization problem is discussed.

To solve the optimization problem described in the previous section using CG approach; first, the independent sets and possible patterns must be identified. Let a pattern refer to a distinct combination of the number of VMs from each type that a server can accommodate. After that, a collection of patterns has been considered for each type of server. Based on server type, T pricing models have been determined. About resource constraints, each pricing problem suggests some configuration candidates to the master problem. If the newly proposed configuration advances the master problem objective, it will be added as a new column to the pattern set. Consequently, the proposed CG technique consists of master and pricing problems. The master problem determines if the explored patterns satisfy the job demand constraints. The pricing models find a new set to feed the master problem. Pricing objective function is, in fact, the reduced cost coefficient of the master problem. The master and pricing problems collaborate until the reduced cost factors (objectives) of all pricing problems are negative, indicating that optimal solution is reached. As discussed earlier, to use CG to solve the optimization problem, the optimization problem should be rewritten in terms of patterns and independent sets and by using new appropriate notations.

Let us define some patterns for each type of server. Thus, for server type $t, j_t \in \{1, , J_t\}$ is defined as a possible pattern. Now let us assume that $x^{j_t}_{r,n}$ represents the scheduling of VM type r of the job n over the server type t by pattern j_t. Hence, there are J_t different ways to put R different VMs of different jobs in server type t. In addition, let us define m_{j_t} as the number of the times pattern j_t of server type t is used in the resource allocation. In other words, m_{j_t} of server type t is required with the configuration j_t in order to serve the jobs in the optimum resource allocation. From the definitions, the following relationships can be written:

$$\widetilde{x}^{j_t}_{n_h} = \sum_{r=1}^{R} x^{j_t}_{r,n_h} \tag{17}$$

$$w^{j_t}_{n_h} = \begin{cases} 1 & if\ x^{j_t}_{n_h} > 0 \\ 0 & x^{j_t}_{n_h} = 0 \end{cases} \tag{18}$$

(17) and (18) are $\forall j_t \in \{1_t, \ldots J_t\}$, $\forall n_h \in \{1_h, \ldots, N_h\}$, $\forall h \in \{1, \ldots, H\}$, $\forall t \in \{1, \ldots T\}$ For employment of CG, the problem should be divided into the master and pricing problems. Consequently, the optimization in the master problem for the centralized problem can be written by:

$$\begin{aligned} \min_{x^{j_t}_{r,n_h}} & \sum_{h=1}^{H} P_h \sum_{n_h=1_h}^{N_h} \sum_{t=1}^{T} \sum_{j_t=1_t}^{J_t} w^{j_t}_{n_h} m_{j_t} + \sum_{t=1}^{T} Q_t \sum_{j_t=1_t}^{J_t} m_{j_t} \\ \text{S.T.} & \sum_{t=1}^{T} \sum_{j_t=1_t}^{J_t} x^{j_t}_{r,n_h} \geq v^r_{n_h} \\ & \sum_{t=1}^{T} \sum_{j_t=1_t}^{J_t} m_{j_t} \leq M_t \\ & \widetilde{x}^{j_t}_{n_h} - w^{j_t}_{n_h} \geq 0 \\ & \theta w^{j_t}_{n_h} - \widetilde{x}^{j_t}_{n_h} \geq 0 \end{aligned} \tag{19}$$

The first term denotes the power consumption of VMs communication while the second one indicates the power consumption of active servers. The first constraint group ensures that the job and VM requirements are satisfied followed by the second group of constraint on number of servers. The last constraint group extracts connectivity variables, $w^{j_t}_{n_h}$, out of the scheduling variables, $x^{j_t}_{r,n_h}$. The pricing problems for each type of t should be written by,

$$\begin{aligned} \min_{x^{j_t}_{r,n_h}} & u^t_{r,n_h} \widetilde{x}^{j_t}_{r,n_h} \\ \text{S.T.} & \sum_{h=1}^{H} \sum_{n_h=1_h}^{N_h} \sum_{r=1} R x^{j_t}_{r,n_h} i^k_t \leq c^k_t \end{aligned} \tag{20}$$

The objective function of pricing problem should be the reduced cost function of the master problem on t^{th} server types. $u^t_{r,n}$ coefficients denote the values of the dual variables of the master problem related to the t^{th} server types. Constraints ensure resource limitations of the servers are met. The candidate patterns will be introduced to the master problem by pricing problems. As long as the reduced cost functions are positive, the algorithm continues. But once the reduced cost functions all together become negative, then pricing issues are terminated and introduce no new candidate to the master problem. Reference [22] applied CG on distributed model end up with;

$$\begin{aligned} \min_{m_{j_t}} & \sum_{h=1}^{H} P_h \sum_{n_h=1_h}^{N_h} \sum_{t=1}^{T} \sum_{j_t=1_t}^{J_t} \\ & \{\sum_{t'=1}^{T} \sum_{m'_{t'}=1_{t'}}^{M_{t'}} m_{j_t} m_{j'_{t'}} x^{j_t}_{r,n_h} x^{j'_{t'}}_{r,n_h} - (m_{j_t} x^{j_t}_{r,n_h})^2\} \\ & + \sum_{t=1}^{T} Q_t \sum_{j_t=1_t}^{J_t} m_{j_t} \end{aligned}$$

Next, the dynamic scheduling using CG is investigated. It is assumed that n^{th}_h job is in the system in the current slot and it will continue to receive service in the next slot. Let $x'^{j_t}_{r,n_h}$, $x^{j_t}_{r,n_h}$ denote the number of type r VMs assigned to this job over the j_t^{th} pattern during the current and next slots, respectively. In this model, the binary variables β_{r,n_h}^{M-tt} are defined to show whether or not r type VMs required by job n_h have migrated from a server, as follows:

$$\beta_{r,n_h}^{m_t} = \begin{cases} 1 & if \sum_{j_t=1_t}^{J_t} (x_{r,n_h}^{j_t} z_{n_h}^{m_t} - x'^{j_t}_{r,n_h} z'^{m_t}_{n_h}) < 0 \\ 0 & Otherwise \end{cases} \tag{21}$$

It is noted that the above summation allows the use of a different pattern at the server as long as it preserves the number of VMs assigned by the original pattern to this job. The additive objective function of the master problem is given by,

$$Min\{\sum_{h=1}^{H} \sum_{n_h=1_h}^{N_h} \sum_{r=1}^{R} G_r \sum_{t=1}^{T} \sum_{m_t=1_t}^{M_t} \beta_{r,n_h}^{m_t} \sum_{j_t=1_t}^{J_t} |x_{r,n_h}^{j_t} - x'^{j_t}_{r,n_h}|\} \tag{22}$$

For dynamic scheduling, Objective (22) should be added to Objective in (19) in the master problem. As in the previous subsection, job scheduling without VM migration can be achieved by setting a very large value for G_r. Finally, similar to the previous subsection, IDBLC has to be applied to linearize Eq. 21.

E. SCHEDULING FOR UNEXPECTED LOAD

Since there is not enough time to set up new servers into the system, active servers should have enough capacity to be able to serve the jobs. Thus, in each time slot, some resources should be reserved for the future unpredicted load. For the unexpected load, the objective is to minimize (1) the extra power consumption related to the servers and (2) communication among the VMs of the jobs. Here, following parameters are defined: $e_{r,n_h}^{m_t}$: The number of type r VMs of unpredicted load of job n_h allocated on server m_t.

$\widetilde{e}_{n_h}^{m_t}$: Total number of VMs required by the unpredicted load of job n_h allocated on server m_t.

Then, the following parameters are defined:

$$\zeta_{n_h}^{m_t} = \begin{cases} 1 & if\ e_{n_h}^{m_t} > 0 \\ 0 & e_{n_h}^{m_t} = 0 \end{cases} \tag{23}$$

Moreover,

$$y_{er}^{m_t} = \begin{cases} 1 & if\ \sum_{h=1}^{H} \sum_{n_h=1_h}^{N_h} e_{n_h}^{m_t} > 0 \\ 0 & \sum e_{n_h}^{m_t} = 0 \end{cases} \tag{24}$$

According to the above definitions, optimization Problem of minimizing the power consumption is given by:

$$\min_{e_{r,n_h}^{m_t}} \sum_{h=1}^{H} P_h \sum_{n_h=1_h}^{N_h} \sum_{t=1}^{T} \sum_{m_t=1_t}^{M_t} \zeta_{n_h}^{m_t}(1 - w_{n_h}^{m_t})$$
$$+ \sum_{t=1}^{T} \sum_{m_t=1_t}^{M_t} y_{er}^{m_t}(1 - y_{m_t})$$

$$\begin{aligned} \text{S.T.} \sum_{h=1}^{H} \sum_{n_h=1_h}^{N_h} (e_{r,n_h}^{m_t} + x_{r,n_h}^{m_t}) i_t^k &\le c_t^k \\ \widetilde{e}_{n_h}^{m_t} - y_{er}^{m_t} &\ge 0 \\ \theta y_{er}^{m_t} - \sum_{h=1}^{H} \sum_{n_h=1_h}^{N_h} \widetilde{xe}_{n_h}^{m_t} &\ge 0 \\ \widetilde{e}_{n_h}^{m_t} - \zeta_{n_h}^{m_t} &\ge 0 \\ \theta \zeta_{n_h}^{h_t} - \widetilde{e}_{n_h}^{m_t} &\ge 0 \end{aligned} \tag{25}$$

The first part of the optimization denotes the communication of fragments of the job caused by the unexpected load while the second part denotes the number of new servers that should be set up only for the unexpected load. It should be noted that the output variables such as $x_{r,n_h}^{m_t}$ of the expected load problem are considered fixed in the optimization problem of the unexpected load. After solving the optimization problem for both expected and unpredicted load, all the results will be sent to an updating module to check the resource constraints and update the variables for the next time slot as it represented in Fig.3.

VI. APPLICATION AND CHALLENGES

There are few concerns with the proposed platform that have to be noticed:

- P_h represents the parameter indicating the power consumption resulted by communication of two VMs allocated to two different servers. This parameter depends on the location of the servers. For instance, the communication between two servers in a rack consumes a different amount of power from those are allocated in two separate racks.
- The scheduling in the proposed platform is done for the entire workload of the DC. Thus, the complexity and the time required to solve the optimization problem becomes a critical factor. Due to the dynamicity of the resource allocation in the DC, it should not take longer than few minutes.

For addressing the first issue, assuming the entire VM requirements of the jobs is less than the resources of a rack, the optimization as mentioned the earlier in the proposed platform has to be solved hierarchically. Our optimization problem is similar to simple balls and urns problem [31]. The problem is how to put the number of balls in the minimum number of the urns. Note that here the balls are the VMs and urns are considered as a modular Power Optimized Datacenter (PoD). After solving the optimization problem on a large scale and allocating jobs to different PoDs, on the smaller scale, each rack is considered as an urn. Finally, on the smallest scale, the PMs are considered as servers inside a rack. In each step, different values should be defined for P_h. In fact, P_h is variant for different types of jobs and has to be calculated based on the previous step.

For the second issue, despite the application of the CG and decomposition methods, computation time, as a

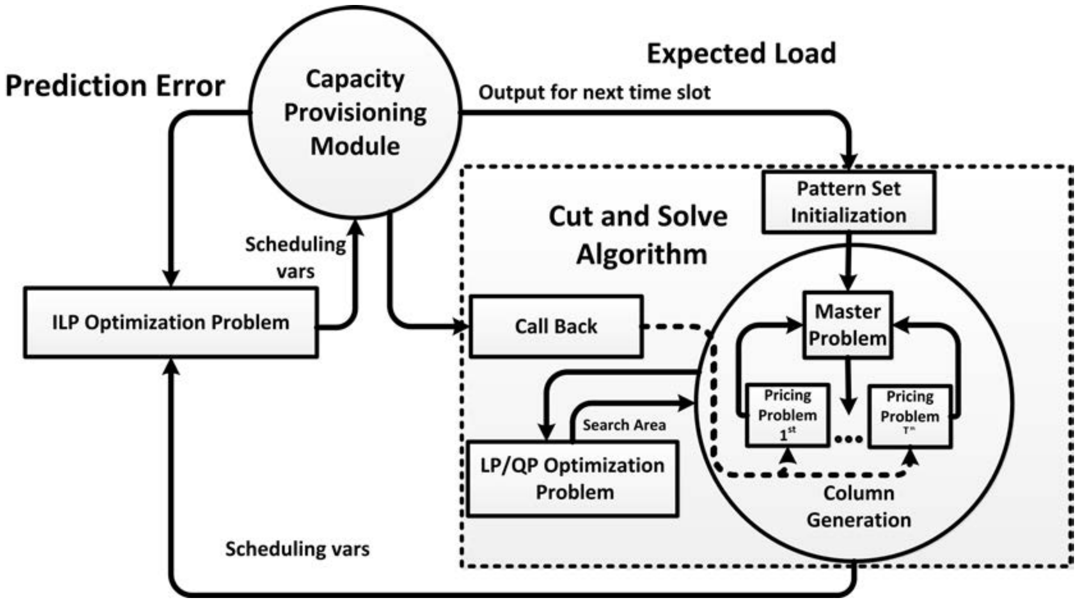


FIGURE 3. **Optimization framework architecture.**

benchmarking constraint, is not satisfied and the optimization problem cannot be solved within a plausible short period. This can be attributed to the complexity order of the problem and a significant number of variables. Hence, Further measures to reduce the computation time are applied. First, in the CG, the offline initialization is implemented to decrease the number of iteration among the master and pricing problems. Given many different types of jobs, first, the offline optimization problem is solved for each server type to obtain the initial server configuration patterns. Moreover, Call-Back method [32] is also applied in the CG optimization problem so that when the time is close to the deadline, the pricing problems would stop searching for better configurations and restricted master problem solve the optimization problem using the existing patterns. Thus, there will be a trade-off between the computation time and optimality. In the case of non-negative reduced cost function of pricing problems, lower the computation time, the less optimal solution. Parallel computation technique should be applied to solve the pricing problems simultaneously.

Finally, cut and solve approach is applied to reduce the complexity of the problem. Cut and solve method performs such that first relaxed problem (LP/QP) is solved. Then, the slice is selected in the searching area, and a new constraint is added to the relaxed problem. The new problem is called sparse problem which provides an incumbent solution. If the incumbent solution solved by CG technique equals to the relaxed problem solution, it is considered as optimal. Otherwise, the slice will be ignored, and a new slice will be selected. So the cuts accumulate with each iteration and finally, solving the sparse problem yields the optimal solution.

The cut and solve mechanism is depicted in Algorithm 1. First, to avoid the switching *on* and *off* the servers, the collection of active servers from the previous slot is pierced as a cut. Here, the term ''critical resource type'' is defined as the most demanding type of resources. In each step, the searching area is accumulated by PMs with the highest capacity on critical resources such that,

$$\Delta = \sum m_t C_t | \sum m_t C_t > \Omega \sum_{h=1}^{H} \sum_{n_h=1_h}^{N_h^{\lambda}+\psi_h^{\lambda}} \sum_{r=1}^{R} v_{n_h}^{r} i_k^{r}$$

Ω is an arbitrary constant which may have some effect on the time required to solve the problem. The higher the value of the Ω, larger the size of the cut. Moreover, the longer time is needed to resolve the optimization problem of each step, and the probability of getting the optimal solution in each cut will be higher.

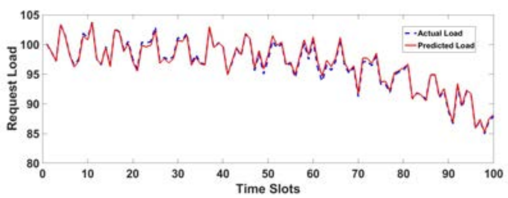
VII. NUMERICAL RESULTS

In this section, some numerical results are presented to evaluate the performance of the estimation module and the schedulers. Time-varying KF is implemented in MATLAB and IBM ILOG CPLEX is used as a platform to model and solve the optimization problems. KF updates the inputs of IBM ILOG CPLEX optimization problem. The results can be applied on OpenStack Liberty through some Nova APIs. Ceilometer module gathers the required information and cinder, and heat modules help to manage the resource allocation of Nova computing instances in the Nova controller node.

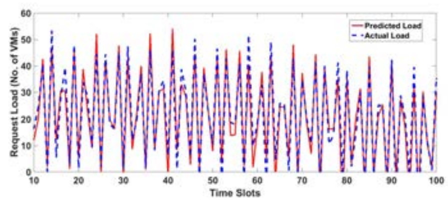
Server instance flavors are selected according to [23, Tables 4.8 and 4.9]. Numerical results plot a performance metric either at a random time or as a function of discrete time. We assume that the number of job types, H, equals

Algorithm 1 Algorithm Cut_and_Solve

procedure –Determining the searching area Solve the LP/QP relaxed problem ();
k = critical resource type();
Sort the server types according to to the value of C_t^k/Q_t.
while $\sum m_t C_t < \Omega \sum_{h=1}^{H} \sum_{n_h=1_h}^{N_h^\lambda+\psi_h^\lambda} \sum_{r=1}^{R} v_{n_h}^r i_k^r$ **do**
Add extra servers according to the sorted list in to the current list of active server
Provoke CG(search area); //to solve the sparse problem
If (CG(search area)== relaxed problem())
return *CG*(*search area*) ;
end while
end procedure



(a)



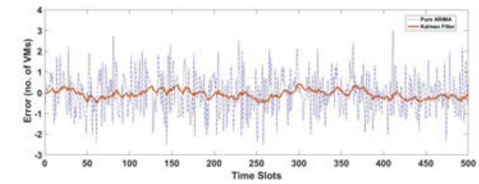
(b)

FIGURE 4. Results of the ARIMA-based KF prediction. (a) h = 1, MMPP with 2 states. (b) h = 5, MMPP with 5 states.

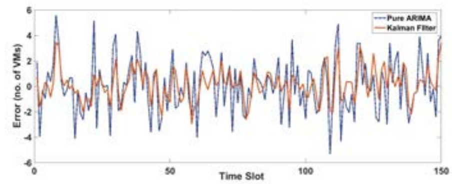
to 5. And new jobs arrive at the cloud DC according to a Markovian Modulated Poisson Process (MMPP), which based on [34] and [35] is a great model for job arrival process, such that the number of states is equal to the job type index.

Fig. 4 shows the acceptable performance of the Estimation module. However, Fig. 4.b indicates that for MMPP with the highest number of states (5) which represents more heterogeneous workloads the performance of KF is degraded a bit. To narrow this issue down and to find the performance gain of the KF, in Fig. 5, the error rate of the KF estimator is compared with the classic ARIMA model. As shown in this figure, the performance of the KF is better than that of the pure ARIMA model such that the average Mean Absolute Error (MAE) of the proposed estimator (1.29) is almost half of the MAE of simple ARIMA model (2.63). Moreover, Fig. 5.b and Fig. 5.c show that even for the most heterogeneous types of arrival rates the ARIMA-based KF part has better tracking performance than the ARIMA.

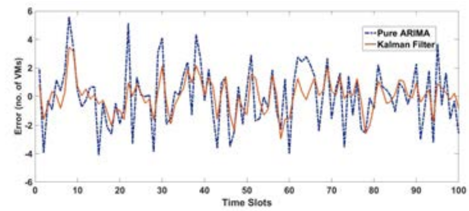
According to [36], one of the best paper in the literature that focused on the optimized placement of VMs to minimize the sum of Network cost and PM-cost is [10].Thus, we compare performance of our optimum resource allocation algorithms with a heuristic scheduling method proposed in [10] that assigns a job to the server with the smallest index number that also has enough idle resources to serve the job. For these results, we assumed $P_h = h * 50W$ and $G_r = r * 20W$. Fig. 6 presents optimal power consumption of the cloud DC as a



(a)

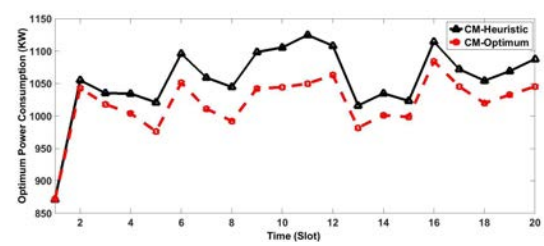


(b)

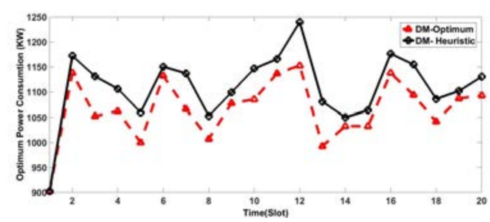


(c)

FIGURE 5. Estimation error. (a) Average error. (b) Error of type 4 jobs. (c) Error of type 5 jobs.



(a)



(b)

FIGURE 6. Optimal and heuristic power consumption of cloud as a function of time. (a) DM. (b) CM.

function of the number of time slots both for centralized and distributed models. We considered optimization of the left-over jobs with individual VM release service discipline. For the VM migration scheme, we also plot consumption of the heuristic proposed in [10]. We note that power consumption varies as a function of time because of the random job arrival process. It may be seen that there is a significant power usage gap (100KW for DM and 68KW for CM) between optimal and heuristic algorithms power consumption which shows value of proposed optimization method.

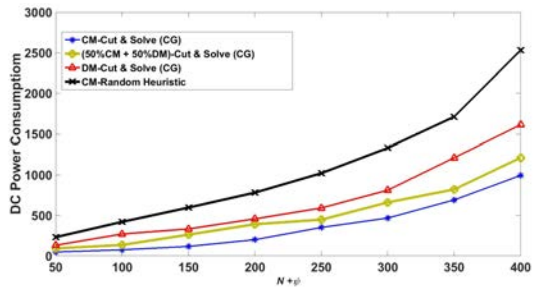


FIGURE 7. Optimum power consumed in DC as a function of total number of jobs in the DC.

Fig. 7 plots the total power consumption as a function of the number of jobs in the cloud DC for optimal and heuristic placement of the VMs of a job in distributed and centralized models. For optimal placement of VMs, results also have been plotted for a hybrid model which included both distributed and centralized jobs. As it shown, there is an enormous power usage gap (1.3MW) between DM optimum resource allocation and heuristic algorithm of CM. And there is also (1.6MW) power usage gap between the total energy consumption of optimal resource planning solution and the heuristic for the half loaded DC which shows we can achieve the optimal solution for power saving by using the proposed CG based cut and solve algorithm of optimal resource allocation method.

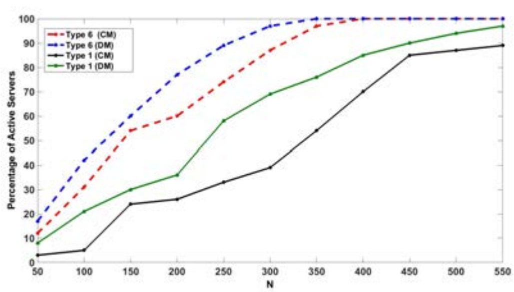


FIGURE 8. Activation rate of the PMs.

Fig. 8 shows the activation percentage of two different types of Dell servers for centralized and distributed models as a function of jobs in the DC. As shown in the figure, type $t = 1, 6$ of servers in CM is much less than the DM. Moreover, as load increases, the aggregation rate of these type of servers in DM is more homogeneous while in the CM, growth rate introduces many random variations compared to DM. It may be caused by a strict dependency of the CM to the network links while DM has less dependency to network connections due to the high communication rate and the higher number of active servers.

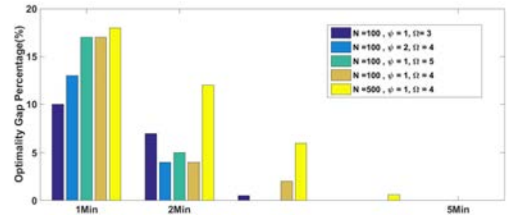


FIGURE 9. Trade off between Computation Time and Optimality.

Fig. 9 shows the trade-off between computation time and optimality. As it mentioned earlier, Call-Back method is employed to end the optimization problem before the deadline. In Fig. 9, optimality gap percentage (the difference between the obtained results and optimal value divided by the optimal value) is presented for a different number of jobs, with different error rates with Ω as a parameter. As it depicted, in less than 3 minutes computation time on a server with two Intel Xeon Processor E5-2660 v2 CPUs and 8x16GB DDR3 (M393B2G70DB0-CMA) RAM, an acceptable near-optimal solution can be achieved.

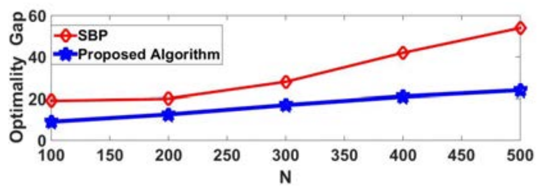


FIGURE 10. Computation Time comparison as a function of DC Scale.

In Fig. 10, the optimality of our proposed solution with work done in [10] named Sorting-based Placement (SBP) under 2 minutes constraint over computation time is compared. As it may seem, the proposed framework is more optimal under different loads (less optimality gap). The impotence of SBP is mainly because of the assumption over heterogeneity of the DC. It is worth mentioning that our proposed platform, despite the higher computation complexity and tight time constraint, it still outperforms SBP.

VIII. CONCLUSION AND FUTURE WORK

In this paper, a platform is proposed for workload prediction and VM placement in cloud computing DC. First, an estimation module was introduced to predict the incoming load of the DC. Then, schedulers were designed to determine the optimal assignment of VMs to the PMs. Column generation method was applied to solve the large-scale optimization problem in conjunction with different algorithms to decrease the complexity and the time to obtain the optimal solution, both on the performance of the proposed platform. Finally, we also investigated the trade-off between optimality and time. Numerical results indicate the proposed platform yields to the optimal solution for a limited time-frame. Our numerical results have shown that our approach explores the optimal solution with an optimality gap of at most 1% in 3 minutes computation time. We have also compared and assessed the performance of our proposed estimation module and state of the art ARIMA estimator. The comparative results prove that our proposed module attains encouraging gain over its peers.

In future work, we think that according to the prediction error, DVFS technique can also be investigated to lessen the processing power consumption. DVFS can be applied to dynamically change voltage and frequency of the cloud servers CPU over the time to save more energy in a sense to compensate the estimation error, higher level of voltage and frequency will be applied.

REFERENCES

[1] A. Berl *et al.*, ''Energy-efficient cloud computing,'' *Comput. J.*, vol. 53, no. 7, pp. 1045–1051, 2010.

[2] L. Minas and B. Ellison, *Energy Efficiency for Information Technology: How to Reduce Power Consumption in Servers and Data Centers*. Hillsboro, OR, USA: Intel Press, 2009.

[3] T. Pering, T. Burd, and R. Brodersen, ''The simulation and evaluation of dynamic voltage scaling algorithms,'' in *Proc. IEEE Int. Symp. Low Power Electron. Design (ISLPED)*, Monterey, CA, USA, 1998, pp. 76–81.

[4] M. Alicherry and T. V. Lakshman, ''Optimizing data access latencies in cloud systems by intelligent virtual machine placement,'' in *Proc. IEEE INFOCOM*, Apr. 2013, pp. 647–655.

[5] A. Gandhi and M. Harchol-Balter, ''How data center size impacts the effectiveness of dynamic power management,'' in *Proc. 49th Annu. Allerton Conf. Commun., Control, Comput.*, Allerton, IL, USA, Sep. 2011, pp. 1164–1169.

[6] A. Gandhi, M. Harchol-Balter, R. Raghunathan, and M. A. Kozuch, ''AutoScale: Dynamic, robust capacity management for multi-tier data centers,'' *ACM Trans. Comput. Syst.*, vol. 30, no. 4, pp. 14–26, Nov. 2012.

[7] M. Alicherry and T. V. Lakshman, ''Network aware resource allocation in distributed clouds,'' in *Proc. IEEE INFOCOM*, Mar. 2012, pp. 963–971.

[8] X. Meng, V. Pappas, and L. Zhang, ''Improving the scalability of data center networks with traffic-aware virtual machine placement,'' in *Proc. INFOCOM*, Mar. 2010, pp. 1–9.

[9] M. H. Ferdaus, M. Murshed, R. N. Calheiros, and R. Buyya, ''Network-aware virtual machine placement and migration in cloud data centers,'' in *Emerging Research in Cloud Distributed Computing Systems*. Hershey, PA, USA: IGI Global Publisher, 2015, pp. 31–42.

[10] X. Li, J. Wu, S. Tang, and S. Lu, ''Let's stay together: Towards traffic aware virtual machine placement in data centers,'' in *Proc. 33rd IEEE Int. Conf. Comput. Commun. (INFOCOM)*, Apr./May 2014, pp. 1842–1850.

[11] V. Chvatal, *Linear Programming*. New York, NY, USA: Macmillan, 1983.

[12] Z. Yang, F. Chu, and H. Chen, ''A cut-and-solve based algorithm for the single-source capacitated facility location problem,'' *Eur. J. Oper. Res.*, vol. 221, no. 3, pp. 521–532, Sep. 2012.

[13] S. Climer and W. Zhang, ''Cut-and-solve: An iterative search strategy for combinatorial optimization problems,'' *Artif. Intell.*, vol. 170, nos. 8–9, pp. 714–738, Jun. 2006.

[14] P. Bodík, R. Griffith, C. Sutton, A. Fox, M. Jordan, and D. Patterson, ''Statistical machine learning makes automatic control practical for internet datacenters,'' in *Proc. Conf. Hot Topics Cloud Comput. (HotCloud)*, San Diego, CA, USA, 2009, Art. no. 12.

[15] A. Krioukov, P. Mohan, S. Alspaugh, L. Keys, D. Culler, and R. H. Katz, ''NapSAC: Design and implementation of a power-proportional Web cluster,'' in *Proc. 1st ACM SIGCOMM Workshop Green Netw., Green Netw.*, New Delhi, India, 2010, pp. 15–22.

[16] A. Verma, G. Dasgupta, T. K. Nayak, P. De, and R. Kothari, ''Server workload analysis for power minimization using consolidation,'' in *Proc. Conf. USE-NIX Annu. Tech. Conf. (USENIX)*, San Diego, CA, USA, 2009, p. 28.

[17] Q. Zhang, M. F. Zhani, Q. Zhu, S. Zhang, R. Boutaba, and J. L. Hellerstein, ''Dynamic energy-aware capacity provisioning for cloud computing environments,'' in *Proc. ACM Int. Conf. Autonomic Comput. (ICAC)*, 2012, pp. 145–154.

[18] Q. Zhang, M. F. Zhani, R. Boutaba, and J. L. Hellerstein, ''Dynamic heterogeneity-aware resource provisioning in the cloud,'' *IEEE Trans. Cloud Comput.*, vol. 2, no. 1, pp. 14–28, Jan./Mar. 2014.

[19] K. You, B. Tang, and F. Ding, ''Near-optimal virtual machine placement with product traffic pattern in data centers,'' in *Proc. IEEE ICC*, Jun. 2013, pp. 3705–3709.

[20] C. Assi, S. Chadi, S. Sebbah, and K. Shaban, ''Towards scalable traffic management in cloud data centers,'' *IEEE Trans. Commun.*, vol. 62, no. 3, pp. 1033–1045, Mar. 2014.

[21] R. N. Calheiros, E. Masoumi, R. Ranjan, and R. Buyya, ''Workload prediction using ARIMA model and its impact on cloud applications' QoS,'' *IEEE Trans. Cloud Comput.*, vol. 3, no. 4, pp. 449–458, Oct./Dec. 2015.

[22] S. Vakilinia, ''Performance modeling and optimization of resource allocation in cloud computing systems,'' Ph.D. dissertation, Dept. Elect. Comput. Eng., Concordia Univ., Montréal, QC, USA, 2015.

[23] S. Vakilinia, M. M. Ali, and D. Qiu, ''Modeling of the resource allocation in cloud computing centers,'' *Comput. Netw.*, vol. 91, pp. 453–470, Nov. 2015.

[24] D. K. Barry, *Web Services, Service-Oriented Architectures, and Cloud Computing: The Savvy Manager's Guide*. San Mateo, CA, USA: Morgan Kaufmann, 2003.

[25] S. Vakilinia, X. Zhu, and D. Qiu, ''Analysis and optimization of big-data stream processing,'' in *Proc. IEEE Globecom*, Washington DC, USA, 2016, pp. 12–18.

[26] A. Thakar and A. Szalay, ''Migrating a (large) science database to the cloud,'' in *Proc. 19th ACM Int. Symp. High Perform. Distrib. Comput.*, 2010, pp. 430–434.

[27] M. T. Özsu and P. Valduriez, *Principles of Distributed Database Systems*. Cambridge, MA, USA: Springer, Feb. 2011.

[28] S. Vakilinia, D. Qiu, and M. M. Ali, ''Optimal multi-dimensional dynamic resource allocation in mobile cloud computing,'' *EURASIP J. Wireless Commun. Netw.*, vol. 2014, no. 1, pp. 1–14, Dec. 2014.

[29] S. J. Julier and J. K. Uhlmann, ''New extension of the Kalman filter to nonlinear systems,'' *Proc. SPIE*, vol. 3068, pp. 182–193, Jul. 1997.

[30] W. Wei, W. Shyong, *Time Series Analysis*. Reading, MA, USA: Addison-Wesley, 1994.

[31] S. Karlin and M.-Y. Leung, ''Some limit theorems on distributional patterns of balls in urns,'' *Ann. Appl. Probab.*, vol. 1, no. 4, pp. 513–538, Nov. 1991.

[32] F. Dabek, N. Zeldovich, F. Kaashoek, D. Mazières, and R. Morris, ''Event-driven programming for robust software,'' in *Proc. 10th ACM Workshop Eur. (SIGOPS)*, 2002, pp. 186–189.

[33] D. Kusic, J. O. Kephart, J. E. Hanson, N. Kandasamy, and G. Jiang, ''Power and performance management of virtualized computing environments via lookahead control,'' *Cluster Comput.*, vol. 12, no. 1, pp. 1–15, Mar. 2009.

[34] D. Bruneo, ''A stochastic model to investigate data center performance and QoS in IaaS cloud computing systems,'' *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 3, pp. 560–569, Mar. 2014.

[35] H. Khazaei, J. Misic, and V. B. Misic, ''Performance of cloud centers with high degree of virtualization under batch task arrivals,'' *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 12, pp. 2429–2438, Dec. 2013.

[36] J. Zhang, H. Huang, and X. Wang, ''Resource provision algorithms in cloud computing: A survey,'' *J. Netw. Comput. Appl.*, vol. 64, pp. 23–42, Apr. 2016.

SHAHIN VAKILINIA (S'07) received the B.Sc. degree in electrical engineering from the University of Tabriz, Tabriz, Iran, the M.Sc. degree in electrical engineering from the Sharif University of Technology, Tehran, Iran, in 2008 and 2010, respectively, and the Ph.D. degree from the Department of Electrical and Computer Engineering, Concordia University, Montreal, QC, Canada, in 2015. He is currently involved in post-doctoral research with the Synchromedia Laboratory, École de technologie supérieure. He has implemented a lot of projects having to do with wireless networks, Internet of Things, and cloud computing systems.

BEHDAD HEIDARPOUR was born in Amol, Iran, in 1985. He received the B.Sc. degree in electrical engineering from the University of Tabriz, Tabriz, Iran, and the M.Sc. degree in electrical engineering from Yazd University, Yazd, Iran, in 2008 and 2011, respectively. He is currently pursuing the Ph.D. degree with the Department of Electrical Engineering, École de technologie supérieure, University of Quebec, Montreal, QC, Canada. His main research interests are economics of data networks, QoS in wireless markets, and cooperative games.

MOHAMED CHERIET (M'83–SM'03) received the Ph.D. degree in computer science from the University of Pierre et Marie Curie. He is currently a Full Professor with the Automation Engineering Department, École de technologie supérieure, University of Quebec. His expertise includes document image analysis, optical character recognition, mathematical models for image processing, pattern classification models, learning algorithms, and perception in computer vision. He co-founded the Laboratory for Imagery, Vision, and Artificial Intelligence and also founded and directs the Synchromedia Consortium (Multimedia Communication in Telepresence) at the University of Quebec.