Future Generation Computer Systems ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Stochastic scheduling for variation-aware virtual machine placement in a cloud computing CPS

Yunliang Chen [a], Xiaodao Chen [a,*], Wangyang Liu [a], Yuchen Zhou [b], Albert Y. Zomaya [c], Rajiv Ranjan [a,d], Shiyan Hu [b]

[a] School of Computer Science, China University of Geosciences, Wuhan, 430074, PR China
[b] Department of Electrical and Computer Engineering, Michigan Technological University, Houghton, MI, 49931, USA
[c] School of Information Technologies, The University of Sydney, Australia
[d] Computing Science at Newcastle University, UK

HIGHLIGHTS

- This paper proposes the two-tier VM placement algorithm that generates the most energy-saving solution with satisfactory feasibility to tolerate resource request variations of VMs.
- The proposed algorithm includes the feasibility driven stochastic VM placement algorithm and the ILP-based variation-unaware VM placement algorithm.
- This work improves the energy cost by 62.7% on average from the algorithm proposed in Gao et al. (2013).
- For the proposed algorithm, the average feasibility reaches 98.0%.

ARTICLE INFO

Article history:
Received 30 November 2016
Received in revised form 11 August 2017
Accepted 7 September 2017
Available online xxxx

MSC:
00-01
99-00

Keywords:
Virtual machine
Cloud computing
Cyber-Physical Systems (CPSs)
Variation-aware
Optimization

ABSTRACT

As the most promising computing paradigm, cloud computing opens up new horizons for the area of high-performance distributed computing. Cyber-physical Systems (CPSs) present novel digital systems, which integrate computation, communication and the control of physical resource. Applied CPSs architecture in cloud computing can provide real-time and scalable resource monitoring and offer time-critical applications. With unrivaled scalability and flexibility, the CPSs based cloud services brings significant convenience to customers in need of elastic computing power. The quality of CPSs based cloud services is, to an large extent, determined by the performance of Virtual Machine (VM) placement algorithm for the data center. VM placement also effect the communication between applications and physical resource distribution in cloud computing CPSs.

The traditional VM placement algorithm is built upon the two-tier architecture. With the presence of multi-media applications, the application level controller cannot accurately quantify the varying amount computing resources required by VMs at runtime. Consequently, lacking accurate resource demand for each VM, controller at the data center level cannot generate the VM placement with satisfactory feasibility. This architecture no longer fits the modern data centers. In this paper, the two tier VM placement framework is proposed to resolve this technical challenge. Our LP-based variation-unaware VM placement algorithm generates the VM placement with minimized energy consumption. On the other hand, our feasibility driven stochastic VM placement (FDSP) algorithm works seamlessly with the LP-based algorithm to achieve desirable feasibility of the placement. Our experimental results show that the LP-based variation unaware VM placement algorithm improves the energy consumption by 15.3% on average from the baseline algorithm. For test cases with resource request variations, the FDSP algorithm saves 15.7% energy cost compared to the "worst case scenario" of the traditional VM placement paradigm. On the other hand, it improves the feasibility by 50.0% compared to the "best case scenario".

* Corresponding author.
E-mail address: cxdao@yahoo.com (X. Chen).

https://doi.org/10.1016/j.future.2017.09.024
0167-739X/

1. Introduction

Cloud computing is reshaping the landscape of high-performance distributed computing [1]. As a state-of-the-art computing paradigm, it offers unparalleled flexibility and computing power to geographically dispersed customers. Commercial cloud service providers, such as Google Compute Engine, Microsoft Azure and Amazon Web Services, are making their cloud computing services affordable and accessible to every individual. Cyber-Physical Systems (CPSs) are the next computing revolution which integrate computing, communication, and storage capabilities with the control of physical world [2]. They are being used in many application domains, including health management, vehicular networks and smart highways, physical infrastructure monitoring [3]. Cloud computing embraced with CPSs as a large-scale distributed computing paradigm can overcome the key challenges in supporting CPSs.

The main challenges for the evolution of CPSs are the reduction of power costs, the enhancement of design, the real-time scalable resource monitoring, the real-time scheduling in hypervisors and high availability [4]. Cloud computing integrates the computation resource, storage and software service to perform real-time parallel computation for massive data [5]. CPSs architecture based cloud computing cooperate seamlessly, the model are shown as Fig. 1. As shown that the architecture of CPSs based cloud computing can be looked as a ecosystem with information flow loops among various physical devices and applications. VM placement in data center controls the physical resource (CPU, memory et al.) distribution, and then changes the performance of applications in a higher layer. The tremendous success of cloud computing lies in the concept of resource sharing and reconfiguration through virtual machines (VM) [6]. The VM is an artificial machine without physical identity, offering computing services directly to end customers. Once the VM completes its computing tasks, the computing resources are returned to the physical server in the data center, and thus can be reused by other customers.

It is the responsibility of the data center to maintain satisfactory Quality of Service (QoS) to all customers at any time [8]. In other words, the VM placer in cloud computing CPSs has to guarantee that each server, hosting multiple VMs, is able to supply sufficient computing resources. On the other hand, in order to reduce operation cost, the cloud computing CPSs tends to minimize energy consumption of the physical servers by shutting down those in the idle state [9]. It requires servers to host more VMs, potentially sacrificing QoS of the cloud service. In this work, the two-tier VM placement algorithm for the cloud data center is proposed. Our algorithm resolves the technical challenge of minimizing the energy consumption of the servers in cloud computing CPSs without sacrificing QoS of the service.

The problem of VM placement optimization has been extensively studied. A number of meta-heuristic algorithms are proposed to optimize the VM placement with multiple objectives [10], including maximizing computing resource usage [11,12], reducing energy consumption [13], and improving network scalability [14,15]. The genetic algorithm (GA) [16] based heuristic is proposed in [17] to model the VM placement problem as the classic grouping problem with the fuzzy-logic controller serving as the cost function in the GA. The traffic-aware VM placer is proposed in [18,19] to minimize the communication cost of the hosting servers without modifying the network architecture. A set of dynamic load distribution policies is proposed in [20] to reduce the electricity energy cost, as an attempt to save operational cost of the data center.

As the state-of-the-art VM placer, the Ant Colony System (ACS) based meta-heuristic is proposed in [21]. The algorithm tries to generate the VM placement solution with minimized energy consumption and computing resource wastage in the data center. According to the problem formulation, the server is mapped to the colony and the VM is mapped to the food source. The assignment of the VM to the server is modeled as the pheromone trail in the ACS algorithm. At each iteration, each pheromone trail is evaluated, among which the most promising ones are selected by the proposed decision-making heuristic. The algorithm terminates when the assignment for all VMs is complete.

Most previous works rely on the two-tier VM placement scheme. The local controller at the application level determines the computing resource requirement on each VM. Subsequently, the global one assigns each VM to the servers at the data center, assuming each VM demands fixed amount of resources. This architecture overlooks the fluctuation of resource requirement of VMs, which is quite common in the practice. Recently, VMs play an important role at the server side with various multi-media applications. The resource required by these applications can be significantly impacted by users from the client side. Meanwhile, the workload of VMs can be also various with different online requires. Thus, when the VMs unexpectedly demand more resources from the server hosting them, the expensive live migration becomes inevitable to avoid the deterioration of the quality of the cloud service. Therefore, the VM placement without considering variations on resource requirement could dramatically increase the operational cost of the data center.

In this paper, we propose an innovative two-tier VM placement algorithm with the flexibility to tolerate the variation on resource requirements of VMs. Our feasibility-driven stochastic VM placement (FDSP) algorithm guarantees the feasibility of the VM placement solution. As the basis of the FDSP algorithm, our ILP-based variation-unaware VM placement algorithm generates the VM placement solution with the minimum energy consumption. Our contributions are as follows.

- We propose the two-tier VM placement algorithm that generates the most energy-saving solution with satisfactory feasibility to tolerate resource request variations of VMs. The two-tier algorithm includes the feasibility-driven stochastic VM placement (FDSP) algorithm and the ILP-based variation-unaware VM placement algorithm. They are seamlessly integrated to achieve our optimization target.
- The ILP-based variation-unaware VM placement algorithm transforms the VM placement problem into the classical LP problem. Given high-performance LP solvers, we improve the energy cost by 62.7% on average from the algorithm proposed in [21]. Meanwhile, the runtime is reduced by 7.6 times for the largest test case.
- For FDSP algorithm, the average feasibility reaches 98.0%. The energy cost is saved by 15.7% on average from the worst case design and the feasibility is improved by 50.0% on average from the best case design.

The remainder of the paper is organized as follows. Section 2 includes the background information and problem formulation. Section 4 introduces the proposed Feasibility Driven Stochastic VM Placement (FDSP) algorithm in cloud computing CPSs. Our experimental results are presented in Section 5. Section 6 concludes the paper.

2. Preliminary

2.1. CPSs based cloud computing

CPSs based Cloud Computing provides the appropriate services to users via the most appropriate devices. This computation model combines the advantages of cloud computing with CPSs [22], and VMs placement links the application layer and physical layer. This

Please cite this article in press as: Y. Chen, et al., Stochastic scheduling for variation-aware virtual machine placement in a cloud computing CPS, Future Generation Computer Systems (2017), https://doi.org/10.1016/j.future.2017.09.024.

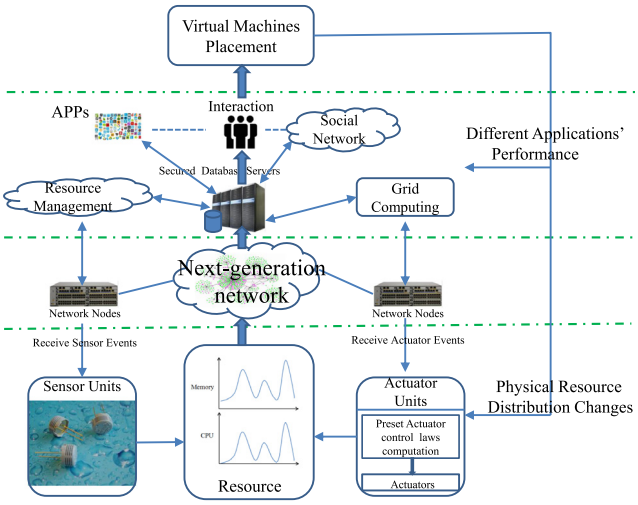


Fig. 1. Cyber-Physical System based Cloud Computing Model [7].

combination not only connect the physical world and virtual computation, but also use the resources efficiently, adapt the environment at every scale, provide rapid scalability and reliable. Cloud computing based on CPSs architecture changes the traditional computation pattern and resource scheduling.

2.2. Virtual cloud environment

The virtual cloud environment is the computing infrastructure which consists of a cluster of networked machines. The computing cluster is physically located at the data center of the cloud service provider [23]. Each machine hosts multiple virtual machines (VMs), artificial machines without physical form directly manipulated by customers. Each VM runs the guest Operating System (OS) and a number of applications of the user. The computing resources requested by the VMs, including CPU, memory and storage, are supplied by the physical server on which the VMs are hosted. Since each VM instance is independent from each other, the computing resource required by VM i can be interpreted as (τ_i, ν_i), in [21], where both τ_i and ν_i are normalized percentage values which denotes the CPU and memory utilization, respectively. For example, if a VM is characterized by the tuple (25%, 30%), it will consume 25% CPU resources and 30% of memory resources of the server.

2.3. Energy consumption model

Studies in [21] and [24] have shown that the power consumption is linearly proportional to the CPU utilization. In the data center, the server can be shut down when its idle. Hence the energy consumption of server j can be defined as

$$E_j = E_{s,j} + C \cdot \sum_i \tau_i \tag{1}$$

in which $E_{s,j}$ is the static energy consumption and C is constant.

3. Problem formulation

This work focuses on resolving the VMs placement problem in cloud computing CPSs such that the energy consumption of data center is minimized. One motivative example is shown in Fig. 2. The resource requirement of VM i is denoted by VM$i(\tau_i, \nu_i)$. The target is to place the five VMs onto a given number of servers, each supplying up to 90% CPU resource and 90% memory resource to the VMs running on it, different placement will result in different users distribution in cloud computing CPSs. Fig. 2(a) shows one valid placement solution, in which each server fulfills the resource demand from each VM. The optimal solution is shown in Fig. 2(b), which places all VMs with two servers. Compared to the non-optimal solution, the optimal one minimizes the number of servers used. Thus it saves approximately 1/3 of energy cost by shutting down one server.

The weakness of traditional VM placement algorithm is their inability to precisely predict the resource requirement of each VM. With the presence of multi-media applications, VMs demand varying amount of resources at run time. Therefore, the practicality of traditional algorithms suffer as the expensive live migration becomes necessary, when the server cannot provide sufficient CPU or memory resource. In cloud computing CPSs, computation abilities will change sharply as physical resource distribution varying. The example of VM placement with resource request variations is shown in Fig. 3. Assuming each VM requires fixed amount of resources, traditional VM placers optimize the VM placement by placing all VMs on the single server, as shown in Fig. 3(a). Since the memory requirement on VM2 decreases from 40% to 30% at runtime, no VM migration is necessary. In the other scenario, shown in Fig. 3(b), VM2 requests 10% more memory at runtime. Consequently, according to the original VM placement, the server cannot provide enough memory resource to the two VMs. In order to maintain the Quality of Service (QoS), either of the two VMs has to be migrated to other servers, incurring significant operational cost.

To overcome the weakness of traditional VM placement algorithms, we formulate the new VM placement problem as follows. Given a set of VMs $V = \{v_1, v_2, v_3, \ldots, v_n\}$ and a set of servers $\mathbb{S} = \{S_1, S_2, S_3, \ldots, S_\Omega\}$, our VM placement algorithm aims at generating the VM placement solution such that the total power consumption E_{total} is be minimized. Our algorithm is subject to the following constraints.

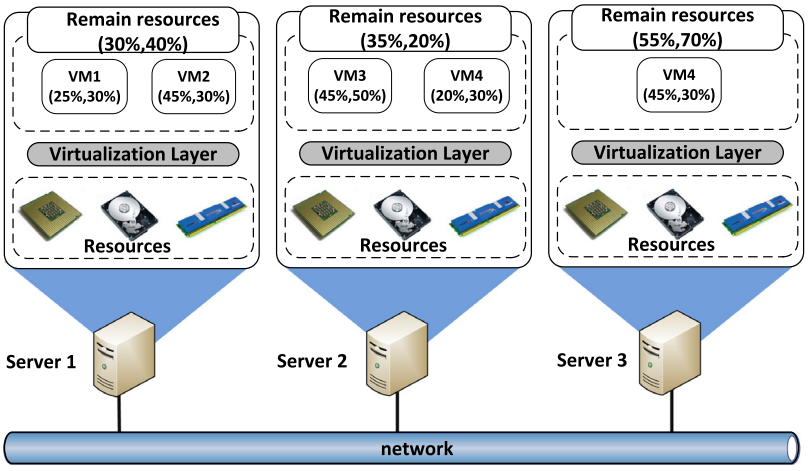
- For each server hosting multiple VMs, the summation of computing resources requested by all VMs may not exceed the maximum amount provided by it.
- The VM placement solution has to achieve desirable feasibility, regardless of variations on the amount of resources required by VMs.

4. Feasibility driven stochastic VM placement (FDSP) algorithm

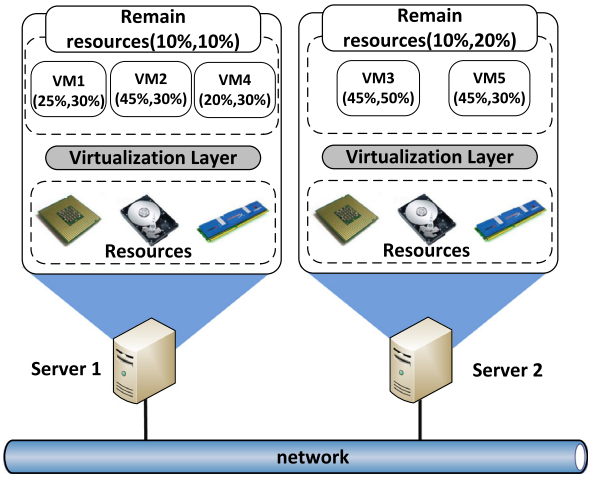
We propose the variation aware VM placement optimization framework for cloud computing CPSs. Our algorithm aims at generating the VM placement solution with the minimum total energy consumption, without sacrificing the Quality of Service (QoS) of the cloud computing CPSs service. On one hand, our algorithm intelligently employs the maximum amount of computing resources on every server for the VMs running on it. On the other hand, it guarantees that, under most circumstances, the resource requirement on every server does not exceed the maximum amount offered by it.

The algorithm flow is shown in Fig. 4. Our algorithm runs in two hierarchical phases. The VM placement optimization phase features the Integer Linear Programming (ILP) formulation of the variation unaware VM placement problem. The QoS optimization phase iteratively queries the most appropriate computing resource requirement for the VMs. At each iteration, the previous optimization phase is triggered and the VM placement solution is evaluated against QoS, in an attempt to obtainthe most feasible

Please cite this article in press as: Y. Chen, et al., Stochastic scheduling for variation-aware virtual machine placement in a cloud computing CPS, Future Generation Computer Systems (2017), https://doi.org/10.1016/j.future.2017.09.024.



(a) VMs placement solution 1.



(b) VMs placement solution 2.

Fig. 2. VM placement examples on two different solutions.

solution. In cloud computing CPSs, this algorithm iteratively adjust the placement of VMs and effect the usage of CPU and Memory, and then optimize the performance of application layer.

4.1. Variation-unaware VM placement

Given the fixed computing resource (CPU and memory) requirement for every VM, we transform the variation-unaware VM placement problem into the ILP optimization problem. Thus it can be efficiently handled by standard Linear Programming (LP) solvers, like LP-Solve and IBM ILOG CPLEX Optimization Studio. The ILP formulation is as follows.

Denote the set of Ω servers as $\mathbb{S} = \{S_1, S_2, S_3, \ldots, S_\Omega\}$ and the set of n VMs as $\mathbb{V} = \{v_1, v_2, v_3, \ldots, v_n\}$. The VM placement solution can be represented by the binary variable set $\mathbb{A} = \{A_{i,j}, 1 \leq i \leq n, 1 \leq j \leq \Omega\}$, in which $A_{i,j}$ indicates that VM v_i is assigned to server S_j. The set of binary variables $\mathbb{B} = \{B_1, B_2, B_3, \ldots, B_\Omega\}$ is introduced to indicate the availability of each server. If B_j is 0, the server S_j is considered unavailable and no VM can be assigned on it. The optimization target is determine $\mathbb{A}$ while minimizing the total energy consumption, E_{total}, under the following constraints.

The total energy constraint defines the total energy consumption as the summation of the energy consumption of each server. Mathematically, it is

$$E_{total} = \sum_{j=1}^{\Omega} E_j, \tag{2}$$

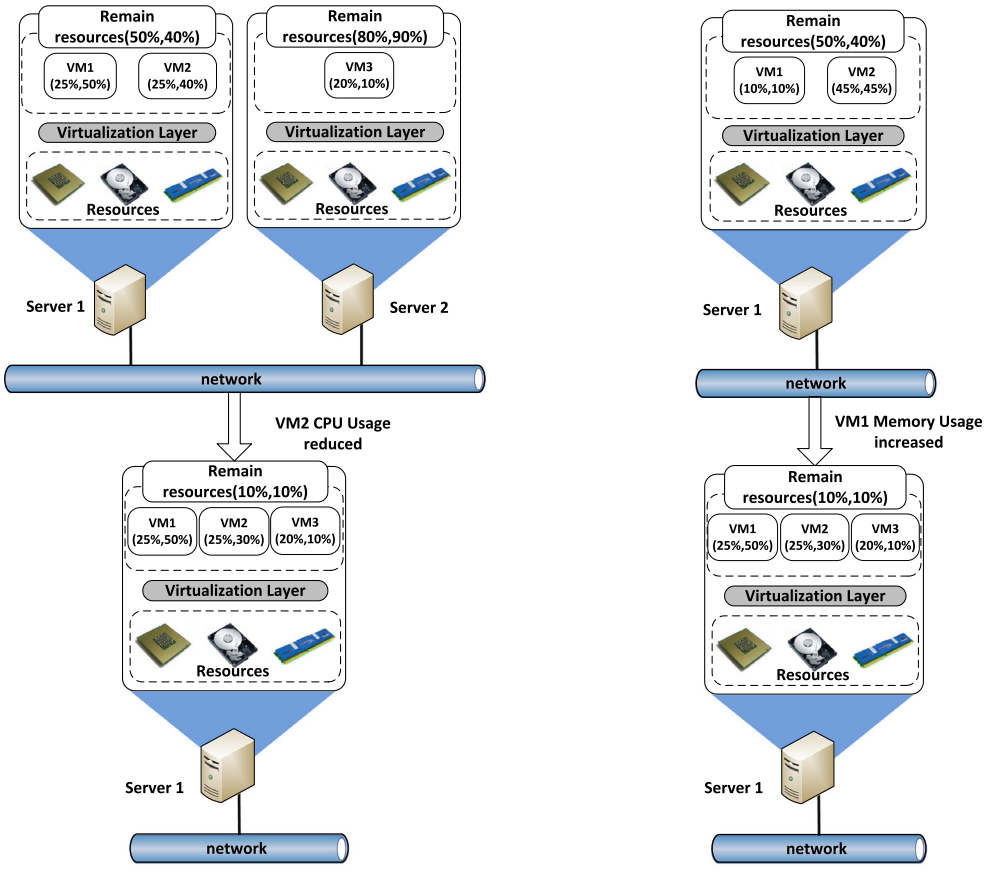
where E_j is the energy consumption of server S_j.

E_j is modeled as the summation of dynamic and static energy consumption of server S_j. The former is proportional to the amount of utilized computing resources while the latter is present only if the server is in use. Denote $p_{i,j}$ as the energy cost contributed by the resource usage of VM v_i on server S_j. The E_j can be formulated as

$$E_j = \sum_{j=1}^{n} A_{i,j} \cdot p_{i,j} + B_j \cdot pd_j \ \forall \ 1 \leq j \leq \Omega \tag{3}$$

To ensure the total CPU resource on each server is not used up by all the VMs running on it, we introduce the CPU resource constraint. It guarantees that the summation of CPU resource required by all VMs on the server does not exceed the maximum amount provided by the server, denoted as C_j for server S_j. Denote $c_{i,j}$ as the

Please cite this article in press as: Y. Chen, et al., Stochastic scheduling for variation-aware virtual machine placement in a cloud computing CPS, Future Generation Computer Systems (2017), https://doi.org/10.1016/j.future.2017.09.024.



(a) CPU usage variations.

(b) Memory usage variations.

Fig. 3. VM usage variations impact the VMs placement.

CPU resource requirement of VM v_i on server S_j. Mathematically the constraint can be represented as

$$\sum_{i=1}^{n} A_{i,j} \cdot c_{i,j} \leq C_j \cdot B_j \ \forall \ 1 \leq j \leq \Omega \tag{4}$$

Denote $m_{i,j}$ as the memory required by VM v_i on server S_j. Similarly, the memory resource constraint can be represented as

$$\sum_{i=1}^{n} A_{i,j} \cdot m_{i,j} \leq M_j \cdot B_j \ \forall \ 1 \leq j \leq \Omega \tag{5}$$

The single assignment constraint is introduced to make sure that every VM is assigned to exactly one server. Thus,

$$\sum_{j=1}^{\Omega} A_{i,j} = 1 \forall \ 1 \leq i \leq n \tag{6}$$

According to the definition of the set $\mathbb{A}$ and $\mathbb{B}$, the following two binary constraints are introduced.

$$A_{i,j} = 0, 1 \ \forall \ 1 \leq i \leq n, 1 \leq j \leq \Omega \tag{7}$$

$$B_j = 0, 1 \ \forall \ 1 \leq j \leq \Omega \tag{8}$$

In practice, The straight forward implementation of the ILP formulation results in [25] prohibitively long run time for the standard ILP solver. To resolve the issue, the iterative rounding technique proposed in is applied to accelerate the ILP solving procedure. According to [25], the binary constraints in Eqs. (7) and (8) are replaced with Eqs. (9) and (10)

$$0 \leq A_{i,j} \leq 1 \ \forall \ 1 \leq i \leq n, 1 \leq j \leq \Omega \tag{9}$$

$$0 \leq B_j \leq 1 \ \forall \ 1 \leq j \leq \Omega \tag{10}$$

In other words, the formulated ILP problem is treated as the LP problem. Initially, every variable is not determined. At each iteration, variables in the set $\mathbb{V} = \{v \mid v \in \mathbb{A} \cup \mathbb{B}, v > 1-\delta\}$ are rounded to 1, where δ is a user defined parameter. If $\mathbb{V} = \varnothing$, the one with the smallest rounding error is rounded to either 1 or 0, depending on its affinity with either 1 or 0. These rounding conditions are feed into the subsequent iterations as additional constraints. Consequently, at least one variable is determined at each iteration so that the entire algorithm converges in at most κ iterations, in which κ is the cardinality of the set $\mathbb{A} \cup \mathbb{B}$.

As defined by the set $\mathbb{A}$ and $\mathbb{B}$, the solution to the ILP problem is directly mapped to the VM placement solution. Therefore, as long as the standard LP solver finds the optimized solution to the ILP problem, the VM placement is obtained, given the fixed CPU and memory requirement for each VM.

4.2. Feasibility driven stochastic VM placement (FDSP)

The ILP based variation-unaware VM placement is not practical as the CPU and memory resource required by each VM constantly changes. Ignoring the variation of computing resource requirement jeopardize the usability of the computing cloud. If the resource requirement is overestimated, the computing infrastructure is forced to operate at high energy cost. The reason is that VMs are sparsely distributed over the servers, some of which could be in the idle state to save power if VMs are optimally placed. On the

Please cite this article in press as: Y. Chen, et al., Stochastic scheduling for variation-aware virtual machine placement in a cloud computing CPS, Future Generation Computer Systems (2017), https://doi.org/10.1016/j.future.2017.09.024.

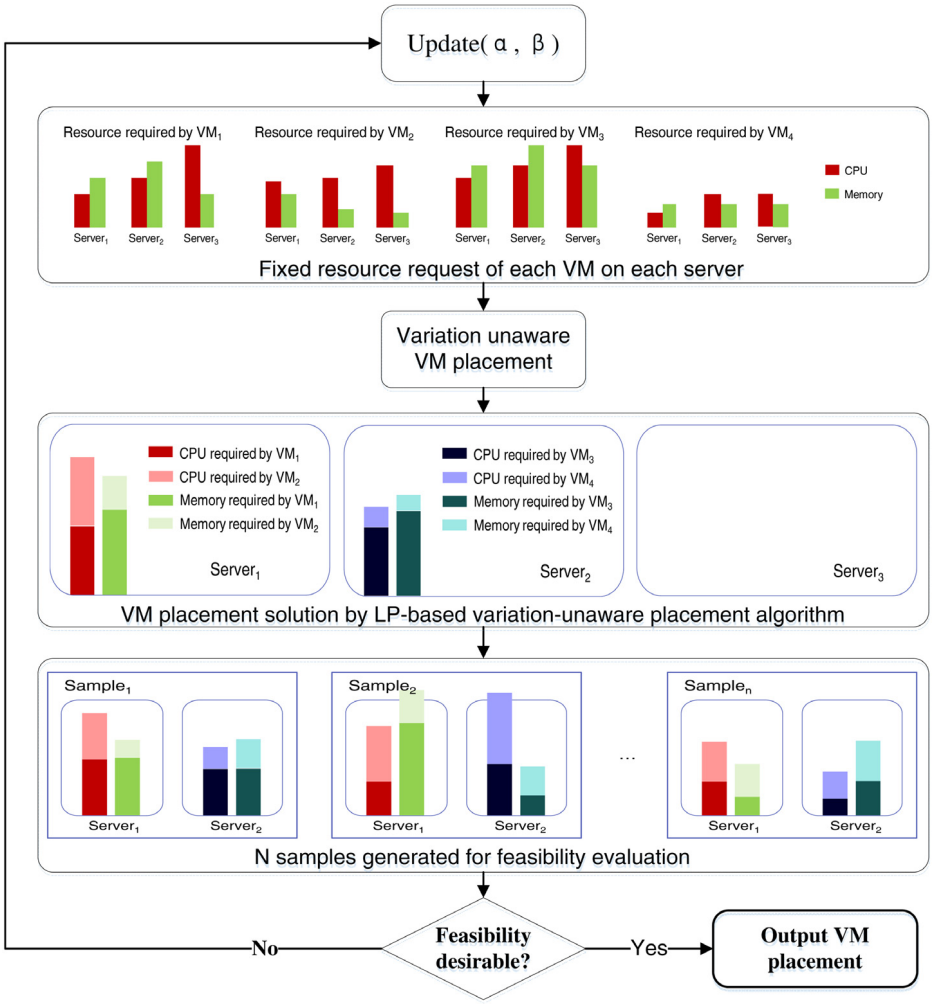


Fig. 4. Basic architecture of our algorithmic framework. Assume there are 3 servers in the data center and 4 VMs to be placed. The FDSP algorithm iteratively updates adaption variables α and β and calls the LP-based variation unaware VM placement algorithm, until the feasibility of the VM placement solution is satisfactory.

contrary, if the resource requirement for VMs is underestimated, QoS of the computing cloud suffers since some servers cannot offer sufficient computing resources to the VMs running on them. The resource estimation is a key technique that improves the utilization of resources in cloud computing CPSs and isolates the applications and physical resources.

To overcome this limitation, we propose the feasibility driven VM placement algorithm that seamlessly integrates the variation-unaware VM placement in cloud computing CPSs. Our algorithm intelligently queries the most appropriate resource requirements for the VMs iteratively such that the energy cost is minimized without degrading the QoS of the computing infrastructure.

Our algorithm is based on the observation that the computing resource required by the VM has limited variation. Denote $cb_{i,j}$ and $cw_{i,j}$ the minimum and maximum CPU resource required by VM v_i on server S_j, respectively. We observe that $cb_{i,j} \leq c_{i,j} \leq cw_{i,j}$ always holds. Similarly, $mb_{i,j} \leq m_{i,j} \leq mw_{i,j}$ holds for memory resources, in which $mb_{i,j}$ and $mw_{i,j}$ are minimum and maximum memory required by VM v_i running on server S_j. For better illustration, two adaption variables α and β are introduced such that $0 \leq \alpha, \beta \leq 1$. $c_{i,j}$ and $m_{i,j}$ can then be represented as

$$c_{i,j} = \alpha \cdot cb_{i,j} + (1-\alpha) \cdot cw_{i,j} \tag{11}$$

$$m_{i,j} = \beta \cdot mb_{i,j} + (1-\beta) \cdot mw_{i,j} \tag{12}$$

As long as α and β are determined, the CPU and memory requirements for all VMs are obtained. Therefore, the ILP based variation-unaware VM placement algorithm can be carried out, given any fixed pair of $\{\alpha, \beta\}$. The target of our algorithm is to find the optimal pair such that the energy cost corresponding to the VM placement is minimized while the feasibility requirement of the placement solution is met. The feasibility of the VM placement is defined as the probability that the computing resource requirements for all VMs are met, for different cases with different amount of resources requested.

The algorithm proceeds as follows. At the beginning, both α and β are initialized to 0. Since $\{\alpha, \beta\}$ pair is fixed, the ILP based variation-unaware VM placement is called to generate a VM placement solution. Subsequently, its feasibility is assessed by our feasibility evaluator using Monte-Carlo method. If the feasibility is satisfactory, the algorithm is terminated and the VM placement solution is returned. Otherwise, α and β are updated using Eqs. (13) and (14).

$$\alpha = \alpha + \delta_\alpha \tag{13}$$

$$\beta = \beta + \delta_\beta \tag{14}$$

δ_α and δ_β are user defined parameters under the constraint $0 < \delta_\alpha, \delta_\beta < 1$. In this way, every VM is assumed to request more resources iteratively. As a consequence, the feasibility requirement is increasingly likely to be met. At the iteration where it is sufficiently satisfactory, the algorithm terminates. Apparently, the algorithm is guided by the feasibility evaluator based on Monte-Carlo method.

Please cite this article in press as: Y. Chen, et al., Stochastic scheduling for variation-aware virtual machine placement in a cloud computing CPS, Future Generation Computer Systems (2017), https://doi.org/10.1016/j.future.2017.09.024.

4.2.1. Feasibility evaluation

Every VM placement solution given the $\{\alpha, \beta\}$ pair is evaluated for feasibility. Monte-Carlo method is used. For each sample, the CPU resource requirement for each VM, $c_{i,j}^{sample}$, is randomly generated with the constraint $cb_{i,j} \leq c_{i,j}^{sample} \leq cw_{i,j}$. If VM v_i is not assigned to server S_j according to the placement, then $c_{i,j}^{sample}$ is set to 0. The memory resource requirement for each VM is generated in the similar manner. After generating k samples, whether each sample is feasible can be easily examined by comparing the total amount of resources required at each server and the maximum amount provided by it. The feasibility of a given VM placement is determined by k_v/k, where k_v is the number of feasible samples among all the generated k samples.

4.2.2. Latin Hypercube Sampling (LHS) method

The run time of the straight forward implementation of the FDSP algorithm is prohibitively high. The reason is that the time-consuming Monte-Carlo method is used in every iteration. To resolve this practicality issue, we introduce the Latin Hypercube Sampling (LHS) method which dramatically speed up the Monte-Carlo method. The reason is that LHS method is able to maintain the statistical performance of the Monte-Carlo method with only limited number of samples. From our observation, it has little impact on our experimental results to replace 5000 Monte-Carlo samples with only 200 samples generated by LHS method.

The essence of LHS method lies in uniform distribution of samples in the entire sample space. In practice, each pair of samples is not allowed to be the same in each dimension. Since they are evenly distributed, they are able to characterize the sample space to the desirable extent, despite the limited number of them. An example of sampling on the 2-D space is shown in Fig. 5 (a_1). Since the dimension is 2, the requirement of LHS method is that each sample covers one unique row and column on the 2-D grid. With only 5 samples, the sample space is sufficiently explored. In contrast, randomly selecting 5 samples results in the situation shown in Fig. 5 (a_2). Apparently, the samples are not uniformly distributed over the 2-D space, with significant amount of space uncovered. The other comparison, with more samples, between LHS method with traditional sampling is shown in Fig. 5 (b_1) and (b_2). It is clear that LHS sampling explores the sample space comprehensively with limited number of samples.

5. Experimental results

Our experiments are performed on a set of heterogeneous computing clusters with different number of nodes/servers. The processors of the machines are fabricated with 70 nm technology. To avoid transient anomalies, the total number of VMs used in the experiment is 500. Each VM carries t computing tasks, where t is a randomly selected integer in the range of [5, 30]. The entire set of 500 VMs are divided into 5 subsets, each containing VMs with similar number of tasks. For each subset, the maximum difference in the number of tasks of VMS is set to be 5. Since the number of tasks of VMs follows uniform distribution, the cardinality of each subset is similar to each other. In our experiments, each test case contains one subset of VMs, to be placed on clusters with various number of nodes/servers.

5.1. ILP based variation-unaware VM placement

To demonstrate the superiority of our ILP based variation-unaware VM placement algorithm, we ran the same set of test cases on our algorithm and the ant colony algorithm proposed in [21], as a baseline algorithm. In this experiment, the computing resource requirements on VMs are assumed to be fixed. Table 1 summarizes energy cost of the generated VM placement and run time of both algorithms. "Task set size" indicates the range in the number of tasks of VMs in each VM subset. "Target platform" suggests the number of nodes/servers in the computing cluster hosting the VMs. "Imp." demonstrates the percentage of improvement from the baseline algorithm. In the table, "CPU" is in second while "Energy" is in kWh. We have the following observations.

Table 1
Comparisons of energy consumption and runtime between the ant colony system algorithm and the proposed LP based algorithm.

Task set size	Target platform	Ant colony system		LP based algorithm		
		CPU (s)	Energy	CPU (s)	Energy	Imp.
5–10	2 nodes	0.12	4872.1	1.80	4016.2	17.6%
11–15	4 nodes	0.82	9668.0	3.29	8002.2	17.2%
16–20	6 nodes	3.60	15390.5	5.32	13234.9	14.0%
21–25	6 nodes	9.96	21441.6	7.47	18557.8	13.4%
26–30	8 nodes	23.34	27949.1	9.71	23970.8	14.2%

- The average improvement on energy cost is 15.3% from the ant colony algorithm, thanks to our ILP formulation. The VM placement problem is transformed into the extensively studied linear programming problem. There exist high-performance ILP solvers that efficiently resolves the ILP formulation with near-optimality.
- The runtime of our algorithm grows linearly with problem size while the baseline algorithm grows exponentially. The reason is still due to the problem transformation. The maturity of LP solvers guarantees the efficiency of our algorithm. It implies the potential of our algorithm to be successfully deployed in the real-world data centers.

5.2. Feasibility driven stochastic VM placement (FDSP)

For assessment of the performance of the feasibility driven VM placement, two greedy algorithms for handling variations on computing resources request from VMs serve as baseline algorithms. One of them assumes that every task on any VMs requires the minimum amount of CPU and memory resources. On the contrary, the other assumes every task requests the maximum amount. All three algorithms, including the proposed one, call the ILP based VM placement algorithm, after variation issue is resolved. The same set of 5 test cases in Section 5.1 are applied on each algorithm, with variation on resource requirement of VMs considered. Table 2 summarizes the statistics of the VM placement solutions generated by each algorithm. In the table, "Best Case Design" and "Worst Case Design" indicate the greedy algorithms assuming minimum and maximum computing resource request from VMs, respectively. We have the following observations.

- "Best Case Design" generates the VM placement with the lowest energy consumption and lowest feasibility. It matches our expectation. The variation-unaware VM placement algorithm assigns most VMs to limited number of servers so that others can be shut down to save energy. As the VMs demands more computing resources, servers with large number of VMs cannot fulfill the requirements. Therefore, in the VM placement evaluation, more than half of samples generated by the Monte-Carlo method are not feasible. It results in the low feasibility for all 5 test cases.
- "Worst Case Design" generates the VM placement with the energy cost as high as 48784.52 kW and 100% feasibility. Since each VM is assumed to consume the maximum amount of resources, the VM placement guarantees that each server provides sufficient computing resources to the VMs running on it, under any circumstances. Therefore, the

Please cite this article in press as: Y. Chen, et al., Stochastic scheduling for variation-aware virtual machine placement in a cloud computing CPS, Future Generation Computer Systems (2017), https://doi.org/10.1016/j.future.2017.09.024.

Table 2
Comparisons of energy consumption, feasibility probability and runtime among the worst case design, the best case design and the proposed FDSP algorithm with varying sizes when tasks are assigned to a cluster system.

Task set	Target	Best Case Deign			Worst Case Deign			Proposed FDSP			
size	platform	Energy	Feasibility	CPU (s)	Energy	Feasibility	CPU (s)	Energy	Feasibility	CPU (s)	Energy Imp.
5–10	2 nodes	2247.9	48.8%	3.40	14313.9	100%	9.40	12075.2	98.0%	44.81	15.6%
11–15	4 nodes	4471.1	48.2%	6.11	28679.9	100%	20.95	24174.6	98.0%	62.67	15.7%
16–20	6 nodes	7431.1	48.0%	9.74	47797.4	100%	23.68	40260.1	98.0%	101.43	15.8%
21–25	8 nodes	10391.1	47.5%	12.89	66993.6	100%	29.88	56396.8	98.0%	142.25	15.8%
26–30	8 nodes	13351.1	47.3%	14.92	86137.8	100%	38.88	72549.9	98.0%	186.80	15.8%
Average		7578.5	48.0%	9.41	48784.52	100%	24.56	41091.32	98.0%	107.59	15.7%

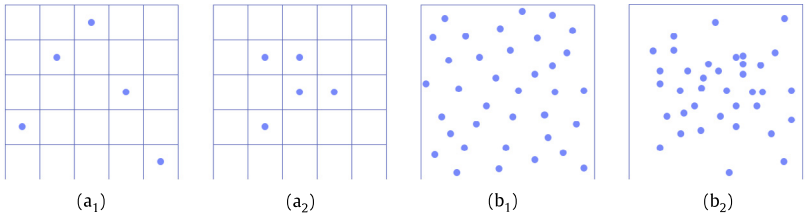


Fig. 5. An example of LHS method on a two-dimensional samples region. (a) Latin Hypercube Sampling. (b) Random Sampling.

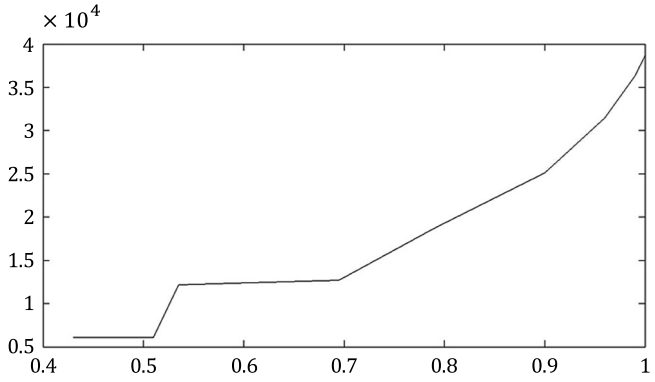


Fig. 6. Feasibility-Energy tradeoff curve for a VM.

feasibility is always 100%, regardless of the variation on the amount of resource requirement of each VM. Apparently, the drawback of this assumption is the inevitable high energy consumption. Since VMs are distributed over a number of servers, only limited number of them can be in the idle state. In other words, the static energy cost contributes significantly to the total energy consumption.

- Our algorithm saves 15.7% energy cost on average from the "Worst Case Design" and improves the feasibility by 50.0% from the "Best Case Design". Our algorithm generates the VM placement that minimizes the energy consumption and maximizes the feasibility. At each iteration, the algorithm assumes that the tasks on each VM consumes more computing resources. In other words, at each iteration, the feasibility of the VM placement solution improves, at the cost of higher energy cost. Eventually, the balance between energy consumption and feasibility is reached. In this way, we obtain the VM placement solution with both two factors optimized.

To better illustrate the trade-off relationship between feasibility and energy consumption of the VM, we plot the Feasibility-Energy curve, as shown in Fig. 6. We have the following observations.

- Apparently, "Best Case Design" and "Worst Case Design' represents the lower left and upper right corner of the curve, respectively. For either case, it is difficult to obtain the VM placement solution with desirable feasibility and energy consumption simultaneously.
- When the feasibility is low, the energy cost increases slowly with it. Therefore, the feasibility of "Best Case Design" can be significantly improved without incurring large energy consumption.
- When feasibility approaches 100%, the energy cost increases almost exponentially with it. That explains why our algorithm improves the energy cost by 15.7% from the "Worst Case Design", at the cost of only "2.0%" decrease in feasibility.

6. Conclusion

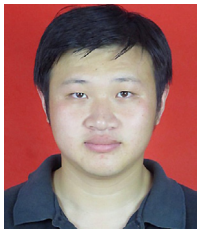
Variation in the amount of resources requested by virtual machines (VMs) remains a technical challenge for VM placement algorithms for cloud computing data centers. Power reduction, real-time scalable resource monitoring and real-time scheduling are main challenges of CPSs. In this paper, we focus on cloud computing CPSs, and propose the two-tier algorithm framework to resolve the challenges mentioned before. Our LP-based variation unaware VM placement algorithm optimizes the VM placement in terms of energy consumption. Our feasibility driven stochastic VM placement (FDSP) algorithm iteratively interact with the LP-based algorithm, to generate the VM placement with satisfactory feasibility. The experimental results show that the LP-based algorithm reduces the energy consumption by 15.3% on average from the baseline algorithm in cloud computing CPSs. The FDSP algorithm achieves the feasibility of 98.0% on average. Compared to "worst case scenario", the energy consumption is reduced by 15.7% on average. Compared to "best case scenario", the feasibility is improved by 50.0% on average.

Acknowledgment

This study was supported in part by the National Natural Science Foundation of China (No. 61501411). Professor Albert Zomaya's work was supported by funding from the Faculty of Engineering & Information Technologies, The University of Sydney, under the Faculty Research Cluster Program.

References

[1] L. Wang, D. Chen, Y. Hu, Y. Ma, J. Wang, Towards enabling cyberinfrastructure as a service in clouds, Comput. Electr. Eng. 39 (2013) 3–14.
[2] R. Rajkumar, I. Lee, L. Sha, J. Stankovic, Cyber-physical systems: the next computing revolution, in: Proceedings of the 47th Design Automation Conference, 2010, pp. 731–736.
[3] S. Bo, X. Zhou, M. Kim, Mixed scheduling with heterogeneous delay constraints in cyber-physical systems, Future Gener. Comput. Syst. 61 (2016) 108–117.
[4] P. Hehenberger, B. Vogel-Heuser, D. Bradley, B. Eynard, T. Tomiyama, S. Achiche, Design, modelling, simulation and integration of cyber physical systems: Methods and applications, Comput. Ind. 82 (2016) 273–289.
[5] Z. Deng, X. Wu, L. Wang, X. Chen, R. Ranjan, A.Y. Zomaya, D. Chen, Parallel processing of dynamic continuous queries over streaming data flows, IEEE Trans. Parallel Distrib. Syst. 26 (2015) 834–846.
[6] L. Wang, G. von Laszewski, A.J. Younge, X. He, M. Kunze, J. Tao, C. Fu, Cloud computing: A perspective study, New Gener. Comput. 28 (2010) 137–146.
[7] Y. Tan, G. S, L.C. P, A prototype architecture for cyber-physical systems, ACM Sigbed Rev. 5 (2008) 1–2.
[8] J. Zhao, L. Wang, J. Tao, J. Chen, W. Sun, R. Ranjan, J. Kołodziej, A. Streit, D. Georgakopoulos, A security framework in G-Hadoop for bigdata computing across distributed Cloud data centres, J. Comput. System Sci. 80 (2014) 994–1007.
[9] Z. Deng, Y. Hu, M. Zhu, X. Huang, B. Du, A scalable and fast OPTICS for clustering trajectory big data, Clust. Comput. 18 (2015) 549–562.
[10] Z. Tang, Y. Mo, K. Li, K. Li, Dynamic forecast scheduling algorithm for virtual machine placement in cloud computing environment, J. Supercomput. 70 (2014) 1279–1296.
[11] C. Isci, J.E. Hanson, I. Whalley, M. Steinder, J.O. Kephart, Runtime demand estimation for effective dynamic resource management, in: Proceedings of the IEEE Network Operations and Management Symposium, 2012, pp. 381–388.
[12] M. Sindelar, R.K. Sitaraman, P. Shenoy, Sharing-aware algorithms for virtual machine colocation, in: Proceedings of the Twenty-Third Annual ACM Symposium on Parallelism in Algorithms and Architectures, 2011, pp. 367–378.
[13] G. Wu, M. Tang, Y.-C. Tian, W. Li, Energy-efficient virtual machine placement in data centers by genetic algorithm, Lecture Notes in Comput. Sci. 7665 (2012) 315–323.
[14] O. Biran, A. Corradi, M. Fanelli, L. Foschini, A. Nus, D. Raz, E. Silvera, A stable network-aware VM placement for cloud systems, in: IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, 2012, pp. 498–506.
[15] M. Wang, X. Meng, L. Zhang, Consolidating virtual machines with dynamic bandwidth demand in data centers, in: IEEE Conference on Information Communications (INFOCOM), 2011, pp. 71–75.
[16] Y. Chen, D. Chen, S.U. Khan, J. Huang, C. Xie, Solving symbolic regression problems with uniform design-aided gene expression programming, J. Supercomput. 66 (2013) 1553–1575.
[17] J. Xu, J.A. Fortes, Multi-objective virtual machine placement in virtualized data center environments, in: Proceedings of the 2010 IEEE/ACM Int'L Conference on Green Computing and Communications & Int'l Conference on Cyber, Physical and Social Computing, IEEE Computer Society, 2010, pp. 179–188.
[18] X. Meng, V. Pappas, L. Zhang, Improving the scalability of data center networks with traffic-aware virtual machine placement, in: INFOCOM, 2010 Proceedings IEEE, IEEE, 2010, pp. 1–9.
[19] Q. Zhang, M. Li, X. Hu, Network traffic-aware virtual machine placement with availability guarantees based on shadows, in :IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, 2014, pp. 542–543.
[20] K. Le, R. Bianchini, J. Zhang, Y. Jaluria, J. Meng, T.D. Nguyen, Reducing electricity cost through virtual machine placement in high performance computing clouds, in: Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis, ACM, 2011, p. 22.
[21] Y. Gao, H. Guan, Z. Qi, Y. Hou, L. Liu, A multi-objective ant colony system algorithm for virtual machine placement in cloud computing, J. Comput. System Sci. 79 (8) (2013) 1230–1242.
[22] E. Simmon, K.-S. Kim, E. Subrahmanian, R. Lee, F. de Vaulx, Y. Murakami, K. Zettsu, R.D. Sriram, A Vision of Cyber-Physical Cloud Computing for Smart Networked Systems, NIST Interagency/Internal Report(NISTIR) 7951.
[23] X. Zhang, Q. Yue, Z. He, Dynamic energy-efficient virtual machine placement optimization for virtualized clouds, Lect. Notes Electr. Eng. 288 (2014) 439–448.
[24] X. Fan, W.-D. Weber, L.A. Barroso, Power provisioning for a warehouse-sized computer, in: ACM SIGARCH Computer Architecture News, vol. 35, ACM, 2007, pp. 13–23.
[25] S. Shah, A. Srivastava, D. Sharma, D. Sylvester, D. Blaauw, V. Zolotov, Discrete vt assignment and gate sizing using a self-snapping continuous formulation, in: Computer-Aided Design, 2005. ICCAD-2005. IEEE/ACM International Conference on, IEEE, 2005, pp. 705–712.



Yunliang Chen received the B.Sc. and M.Eng degree from China University of Geosciences, and the Ph.D. degree from Huazhong University of Science and Technology, China. He is currently an Associate Professor with the School of Computer Science, China University of Geosciences, Wuhan, China. His research interests include data management, cloud computing and high-performance computing.

Xiaodao Chen received the B.Eng. degree in telecommunication from the Wuhan University of Technology, Wuhan, China, in 2006, the M.Sc. degree in electrical engineering from Michigan Technological University, Houghton, USA, in 2009, and the Ph.D. in computer engineering from Michigan Technological University, Houghton, USA, in 2012. He is currently an Associate Professor with School of Computer Science, China University of Geosciences, Wuhan, China. His research interests include Design Automation for petroleum system, High Performance Computing and Optimization.

Wangyang Liu received the B.Sc. and M.Eng degree from China University of Geosciences. His research interests include data management and high-performance computing.

Yuchen Zhou received the B.S. degree in microelectronics from Hefei University of Technology, Hefei, China, in 2010. He is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, Michigan Technological University, Houghton, MI, USA.

Albert Zomaya received the Ph.D. degree from the Department of Automatic Control and Systems Engineering, Sheffield University in the United Kingdom. He is currently the chair professor of High Performance Computing & Networking and Australian Research Council Professorial Fellow in the School of Information Technologies, The University of Sydney. He is also the director of the Centre for Distributed and High Performance Computing which was established in late 2009. He held the CISCO Systems chair professor of Internet working during the period 2002–2007 and was also the head of school for 2006–2007 in the same school. Prior to his current appointment he was a full professor in the School of Electrical, Electronic and Computer Engineering at the University of Western Australia, where he also led the Parallel Computing Research Laboratory during the period 1990–2002. He served as an associate-, deputy-, and actinghead in the same department, and held numerous visiting positions and has extensive industry involvement. He is a fellow of the IEEE.

Rajiv Ranjan is a reader in the School of Computing Science at Newcastle University. He is an internationally established scientist with about 160 publications and expertise in cloud computing, big data, and Internet of Things. Before moving to Newcastle University, he was Julius Fellow, Senior Research Scientist and Project Leader in the Digital Productivity and Services Flagship of Commonwealth Scientific and Industrial Research Organization. Ranjan received a Ph.D. in computer science from the University of Melbourne. He serves on the editorial boards of top quality international journals including IEEE

Please cite this article in press as: Y. Chen, et al., Stochastic scheduling for variation-aware virtual machine placement in a cloud computing CPS, Future Generation Computer Systems (2017), https://doi.org/10.1016/j.future.2017.09.024.

Transactions on Computers, IEEE Transactions on Cloud Computing and IEEE Cloud Computing.

Shiyan Hu received his Ph.D. in Computer Engineering from Texas A&M University in 2008. He is an Associate Professor at Michigan Tech. where he is Director of Center for Cyber-Physical Systems and Associate Director of Institute of Computer and Cybersystems. He has been a Visiting Professor at IBM Research (Austin) in 2010, and a Visiting Associate Professor at Stanford University from 2015 to 2016. His research interests include Cyber-Physical Systems, Cybersecurity, Computer-Aided Design of VLSI Circuits, and Embedded Systems, where he has published more than 100 refereed papers.

He is an ACM Distinguished Speaker, an IEEE Computer Society Distinguished Visitor, an invited participant for U.S. National Academy of Engineering Frontiers of Engineering Symposium, a recipient of National Science Foundation (NSF) CAREER Award, a recipient of ACM SIGDA Richard Newton DAC Scholarship (as the faculty advisor), and a recipient of JSPS Faculty Invitation Fellowship. He is the Chair for IEEE Technical Committee on Cyber-Physical Systems. He serves as an Associate Editor for IEEE Transactions on Computer-Aided Design, IEEE Transactions on Industrial Informatics, and IEEE Transactions on Circuits and Systems. He is also a Guest Editor for 7 IEEE/ACM Transactions such as IEEE Transactions on Computers and IEEE Transactions on Computer-Aided Design. He has served as chairs, TPC chairs, TPC track chairs and TPC members for numerous conferences.

Please cite this article in press as: Y. Chen, et al., Stochastic scheduling for variation-aware virtual machine placement in a cloud computing CPS, Future Generation Computer Systems (2017), https://doi.org/10.1016/j.future.2017.09.024.